Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων

Σύστημα προσομοίωσης δικτύου ομότιμων κόμβων με σχήματα RDF

Διπλωματικη Εργασια

του

ΑΛΕΞΑΝΔΡΟΥ Α. ΜΠΑΞΕΒΑΝΗ

Επιβλέπων: Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

Εργαστηριο Συστηματων Βασεων Γνωσεων και Δεδομενων

Αθήνα, Ιούλιος 2005

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων

Σύστημα προσομοίωσης δικτύου ομότιμων κόμβων με σχήματα RDF

Διπλωματικη Εργασια

του

ΑΛΕΞΑΝΔΡΟΥ Α. ΜΠΑΞΕΒΑΝΗ

Επιβλέπων: Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 11η Ιουλίου 2005.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Τιμολέων Σελλής	Ιωάννης Βασιλείου	Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2005

(Υπογραφή)

.....

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

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων

Η παρούσα εργασία στοιχειοθετήθηκε με το σύστημα LaTeX.

Αλέξανδρος Μπαξεβάνης, *Σύστημα προσομοίωσης δικτύου ομότιμων κόμβων, με σχήματα RDF*, Διπλωματική Εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών, Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων, 11 Ιουλίου 2005.

Σελίδες: 87

Ευχαριστίες

Ευχαριστώ τον επιβλέποντα καθηγητή μου κ. Τίμο Σελλή καθώς και τον Δρ$^{\alpha}$. Θοδωρή Δαλαμάγκα, ερευνητή στο Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων για την υποστήριξή τους κατά τη διάρκεια της εκπόνησης της διπλωματικής.

Η διπλωματική αυτή αφιερώνεται σε όλους τους φίλους που με στήριξαν, ο καθένας με τον τρόπο του, στην πορεία μου στο Εθνικό Μετσόβιο Πολυτεχνείο: στο Δημήτρη Κουζή-Λουκά, στη Μαρίλια Κανίνια και πάνω απ' όλα στη σύντροφό μου Ania Mendrek.

Περίληψη

Τα δίκτυα ομότιμων κόμβων, μια από τις πιο σημαντικές τεχνολογίες στην ιστορία του διαδικτύου, προσφέρουν έναν αποτελεσματικό τρόπο κατανεμημένης αποθήκευσης και διαχείρισης της πληροφορίας, μειώνοντας το κόστος διαχείρισης ενός δικτύου και ελαχιστοποιώντας τις ζημιές σε περίπτωση αστοχίας σε ένα ή περισσότερα σημεία του. Μια άλλη σημαντική τεχνολογία, το πλαίσιο RDF, παρέχει έναν ευέλικτο και εκφραστικό τρόπο φορητής αναπαράστασης κάθε είδους πληροφορίας. Ο συνδυασμός των τεχνολογιών αυτών οδηγεί σε μια νέα γενιά δικτύων ομότιμων κόμβων βασισμένων σε σχήματα.
Το σύστημά μας παρέχει μια πλήρη πλατφόρμα εξομοίωσης, η οποία διευκολύνει την ανάπτυξη και τη δοκιμή διαφορετικών αλγορίθμων απάντησης ερωτημάτων και γενικά αλγορίθμων που αφορούν δίκτυα ομότιμων κόμβων βασισμένα σε σχήματα RDF. Ο εξομοιωτής περιέχει έτοιμα υποσυστήματα τα οποία αναλαμβάνουν τη σύνδεση με το δίκτυο ομότιμων κόμβων και χειρίζονται τα δεδομένα RDF παρέχοντας προς το χρήστη εύχρηστες διεπαφές. Ο πειραματισμός διευκολύνεται επιπλέον από έναν σύνολο γεννητριών τυχαίων σχημάτων, τοπολογιών δικτύου και ερωτημάτων, με βάση τα αποτελέσματα των οποίων ο χρήστης μπορεί να δοκιμάσει τους αλγορίθμους κάτω από πολλές διαφορετικές συνθήκες.
Ιδιαίτερη σημασία έχει δοθεί στην απλότητα της εγκατάστασης και στην αυτοματοποίηση των σεναρίων εξομοίωσης. Τέλος, ας σημειωθεί ότι η πλατφόρμα εξομοίωσης μπορεί να αποτελέσει με ελάχιστες τροποποιήσεις τη βάση για τη δημιουργία μιας πραγματικής κατανεμημένης εφαρμογής, μιας και το υποσύστημα διαχείρισης του δικτύου ομότιμων κόμβων βασίζεται στην τεχνολογία JXTA, η οποία επιτρέπει την δημιουργία επεκτάσιμων δικτύων.

Λέξεις Κλειδιά

Δίκτυα ομότιμων κόμβων, RDF, RDF Schema, πλατφόρμα εξομοίωσης, JXTA

Abstract

Peer-to-peer networks, one of the most important technologies in the history of the Internet, offer an effective method for distributed storage and processing of information, reducing the cost of network management and minimizing loss in case of failure in one or more nodes. Another important technology, the RDF Framework, offers a flexible, expressive and portable representation of all kinds of information. The combination of these two technologies leads to a new generation of peer-to-peer network based on schemas.
Our system supplies a complete simulation platform that facilitates development and testing of different query answering algorithms and, in general, algorithms useful with RDF Schema-based peer-to-peer networks. The simulator contains ready-made subsystems that can handle the connection with a peer-to-peer network and can process RDF data, presenting the user with simple APIs. Experimentation is further facilitated by a number of generators for random schemas, topologies and queries, that can help users test their algorithms under many different circumstances.
Simplicity of installation and automation of simulation scenarios were among the most important design goals. Finally, we must also note that, with few modifications, our platform can be used to create a real-world fully distributed application, since the peer-to-peer layer is based on JXTA, a technology that supports scalable peer-to-peer networks.

Keywords

Peer-to-peer networks, RDF, RDF Schema, simulation platform, JXTA

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

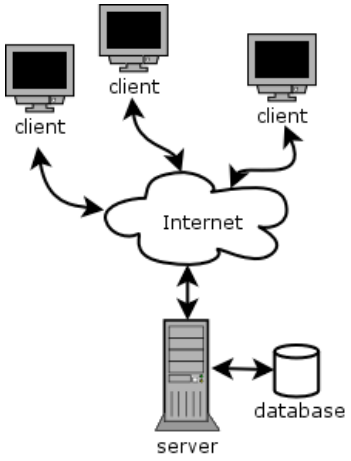
Εισαγωγή

Μια από τις μεγαλύτερες προκλήσεις για την περαιτέρω εξάπλωση του διαδικτύου είναι η εύρεση κοινού τόπου ανάμεσα στην κανονικότητα (normalization) και την αυτό-οργάνωση (self-organization)[Zhu05]. Από τη μια πλευρά, η κανονικότητα εξασφαλίζει την τάξη, τη σταθερότητα και την αρμονική συνεργασία μεταξύ των στοιχείων ενός συστήματος. Από την άλλη, η αυτό-οργάνωση επιτρέπει τη βέλτιστη αξιοποίηση των διαθέσιμων πόρων μέσω της ανάπτυξης αυτόνομων προσαρμοστικών συστημάτων.

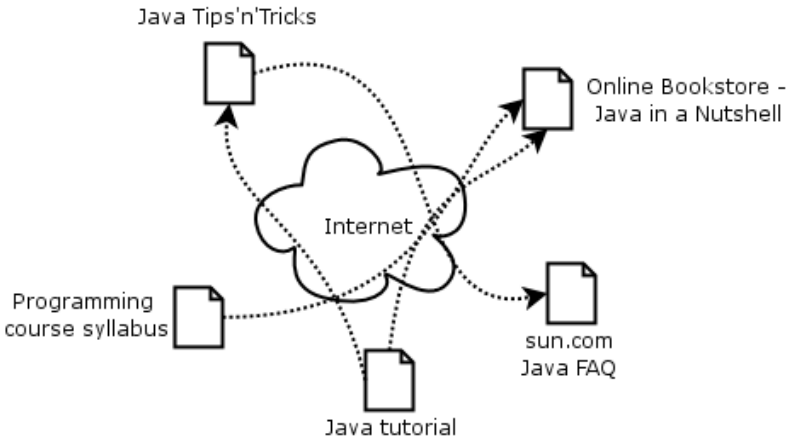
Είναι ευρέως γνωστό το γεγονός ότι το διαδίκτυο είναι ένα ανθεκτικό δίκτυο, ότι δηλαδή υπάρχει πλεονασμός (redundancy) τέτοιος που επιτρέπει τη συνεχή λειτουργία του δικτύου ακόμη και μετά από έναν αριθμό καταστροφικών γεγονότων. Αυτό μάλιστα ήταν ένας από τους αρχικούς στόχους της ανάπτυξης του διαδικτύου, όταν ξεκίνησε ως ερευνητικό έργο με στρατιωτική χρηματοδότηση.

Πρέπει όμως να επισημανθεί ότι η ανθεκτικότητα αυτή αφορά το επίπεδο των διασυνδέσεων δικτύου, και μόνο παράπλευρα επηρεάζει τη δυνατότητα αδιάλειπτης πρόσβασης στην πληροφορία που είναι διαθέσιμη στο διαδίκτυο. Ακόμη και σήμερα, είναι πολύ συχνό το φαινόμενο να αποθηκεύεται ένας μεγάλος όγκος δομημένης πληροφορίας σε ένα κεντρικό σημείο (πιθανότατα με χρήση κάποιας βάσης δεδομένων), η δε διαχείριση και ενημέρωση να γίνεται και πάλι κεντρικά (βλ. Σχήμα 1.1). Στην περίπτωση αυτή υπάρχει ένα μοναδικό σημείο όπου μπορεί να παρουσιαστεί σφάλμα (single point of failure) που θα επηρεάσει τη διαθεσιμότητα του δικτύου. Η μόνη πρακτική λύση για το πρόβλημα αυτό είναι η αυτοματοποιημένη δημιουργία αντιγράφων (replication).

Εκτός βέβαια από τη διευκόλυνση της πρόσβασης σε κεντρικές βάσεις δεδομένων, το διαδίκτυο οδήγησε και στη δημιουργία μιας νέας μορφής οργάνωσης της πληροφορίας, αυτό που σήμερα ονομάζεται Παγκόσμιος Ιστός (World Wide Web). Ο Παγκόσμιος Ιστός αποτελείται ουσιαστικά από μια συλλογή αδόμητης πληροφορίας, τυποποιημένης μόνο ως προς τον τρόπο εμφάνισης και πλοήγησης (γλώσσα HTML και πρωτόκολλο HTTP). Είναι προφανές ότι οι πληροφορίες που περιέχονται στον Παγκόσμιο Ιστό προορίζονται κυρίως για ανάγνωση από ανθρώπινα όντα παρά για συστηματική επεξεργασία. Σε κάποιες περιπτώσεις μάλιστα, μπορεί να είναι ημιτελείς ή ακόμη και λανθασμένες. Αυτό όμως που χαρακτηρίζει τον Παγκόσμιο Ιστό είναι η δομή υπερκειμένου (hypertext) μέσω της οποίας μπορούν να



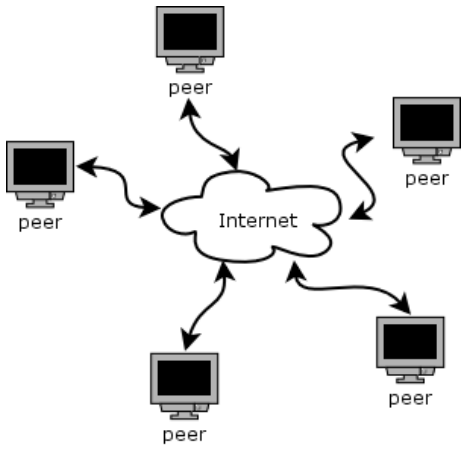
Σχήμα 1.1: Αρχιτεκτονική client-server



Σχήμα 1.2: Διασυνδεδεμένες ιστοσελίδες

συνδεθούν και να σχηματίσουν διάφορες (αυθαίρετες) δομές από σχετικές πληροφορίες. Οι σχετικές αυτές πληροφορίες στην πλειοψηφία των περιπτώσεων φιλοξενούνται σε διαφορετικούς εξυπηρετητές (servers) με αποτέλεσμα μια τοπική βλάβη να μη μπορεί να μειώσει σημαντικά την αξία της παρεχόμενης πληροφορίας. Ένα υποθετικό σύνολο συνδεδεμένων ιστοσελίδων φαίνεται στο Σχήμα 1.2.

Η τελευταία σημαντική εξέλιξη στο χώρο του διαδικτύου είναι τα Δίκτυα Ομότιμων Κόμβων (Peer-to-Peer ή για συντομία P2P δίκτυα). Με τη νέα αυτή θεώρηση, υποβιβάζεται ή καταργείται εντελώς ο ρόλος ενός κεντρικού εξυπηρετητή και κάθε κόμβος που συμμετέχει στο δίκτυο αλληλεπιδρά ελεύθερα με όλους τους υπόλοιπους κόμβους (βλ. Σχήμα 1.3). Δίκτυα με τη μορφή αυτή έχουν χρησιμοποιηθεί ευρέως, κυρίως όμως για τη μεταφορά και κατανεμημένη αποθήκευση αρχείων (μουσική, εικόνες, λογισμικό) και σε κάποιο βαθμό για την ανταλλαγή μηνυμάτων (Instant Messaging). Ιδιαίτερο ενδιαφέρον παρουσιάζουν όμως οι αλγόριθμοι και οι τεχνικές που έχουν αναπτυχθεί στο πλαίσιο των δικτύων P2P και οι οποίες μπορούν να γενικευθούν στην οργάνωση ενός ομότιμου δικτύου για οποιαδήποτε χρήση.



Σχήμα 1.3: Δίκτυο Ομότιμων Κόμβων (p2p)

Απώτερος στόχος, όπως αναφέρθηκε και στην αρχή του κειμένου, είναι τελικά να συνδυαστεί η υψηλή διαθεσιμότητα και η ευελιξία των αδόμητων P2P δικτύων με τη δυνατότητα για αναζήτηση και διακίνηση δομημένης πληροφορίας. Η ενσωμάτωση δομημένης πληροφορίας στον Παγκόσμιο Ιστό αποτελεί εξάλλου βασική προτεραιότητα της πρωτοβουλίας για ένα Σημασιολογικό Ιστό (Semantic Web) [BLHL01].

1.1 Αντικείμενο της διπλωματικής

Αντικείμενο της παρούσας εργασίας είναι η σχεδίαση και ανάπτυξη ενός πρότυπου δικτύου ομότιμων κόμβων με κατανεμημένη αποθήκευση δομημένης πληροφορίας. Στα πλαίσια του δικτύου αυτού, κάθε κόμβος μπορεί να προωθεί κατάλληλα διαμορφωμένα ερωτήματα προς άλλους κόμβους και να παράγει μια απάντηση που θα περιέχει πληροφορίες από όλους τους ερωτηθέντες κόμβους που περιέχουν τις αναζητηθείσες πληροφορίες. Επιπλέον, κάθε κόμβος έχει την ελευθερία να αποθηκεύει τις πληροφορίες με ελαφρά διαφορετική δομή, πάντα όμως συμβατή με τη δομή που χρησιμοποιούν οι υπόλοιποι κόμβοι. Σε πραγματικά σενάρια διαπιστώνουμε συχνά ότι οι συμμετέχοντες στο δίκτυο είναι αδύνατον να πεισθούν να παρέχουν δομημένες πληροφορίες με την ίδια ακριβώς δομή. Για το λόγο αυτό είναι σημαντικό να υποστηρίζει το δίκτυό μας πολλές διαφορετικές (αλλά συμβατές) δομές.

Το σύστημα έχει αναπτυχθεί με σκοπό τον πειραματισμό και τη δυνατότητα επεκτάσεων στο μέλλον. Προσφέρει λειτουργίες που βοηθούν στην αυτοματοποίηση των μετρήσεων (benchmarking) και στην εκσφαλμάτωση (debugging).

Ένα παράδειγμα της εφαρμογής ενός τέτοιου συστήματος θα μπορούσε να είναι η ταυτόχρονη αναζήτηση σε ένα σύνολο από διαδικτυακά βιβλιοπωλεία. Κάθε βιβλιοπωλείο θα συμμετέχει στο δίκτυο με έναν κόμβο παρέχοντας τις πληροφορίες που διαθέτει για τα βιβλία του (π.χ. Τίτλος, Συγγραφέας, Εκδότης) αλλά και άλλες πληροφορίες (π.χ. σχόλια από αγοραστές). Στο δίκτυο αυτό θα μπορούσαν να συμμετάσχουν λογοτεχνικά περιοδικά (παρέχοντας λ.χ. κριτικές βιβλίων) ή και άλλοι ερασιτεχνικοί ιστότοποι που περιέχουν, για παράδειγμα, βιογραφίες συγγραφέων.

1.2 Οργάνωση του τόμου

Η εργασία αυτή χωρίζεται σε 7 κεφάλαια:

- Στο 2^o κεφάλαιο παρουσιάζονται οι βασικές τεχνολογίες (σχήματα RDF, δίκτυα P2P) πάνω στις οποίες στηρίζεται η εργασία, γίνεται μια σύνοψη σχετικών εργασιών και αναφέρεται η συνεισφορά της εργασίας μέσα στο παραπάνω πλαίσιο.
- Στο 3^o κεφάλαιο παρουσιάζεται το θεωρητικό υπόβαθρο των σχημάτων RDF και δίνονται οι ορισμοί για τη συμβατότητα RDF σχημάτων.
- Στο 4^o κεφάλαιο περιγράφεται η ανάλυση και η σχεδίαση του συστήματος και καθορίζεται η βασική αρχιτεκτονική με τα αντίστοιχα υποσυστήματα προς υλοποίηση
- Στο 5^o κεφάλαιο δίνονται οι λεπτομέρειες υλοποίησης του συστήματος, οι σημαντικότεροι αλγόριθμοι και ο σκελετός του κώδικα του προγράμματος.
- Στο 6^o κεφάλαιο παρουσιάζονται τα αποτελέσματα δοκιμών του συστήματος καθώς και διάφορα σενάρια χρήσης.
- Στο 7^o κεφάλαιο απαριθμούνται τα συμπεράσματα της εργασίας καθώς και προτάσεις για μελλοντικές επεκτάσεις.

Στο τέλος του τόμου βρίσκεται η σχετική βιβλιογραφία που αναφέρεται στο κείμενο.

Κεφάλαιο 2

Περιγραφή Θέματος

2.1 Θεωρητικό υπόβαθρο

2.1.1 Δίκτυα ομότιμων κόμβων (p2p)

Από τους κεντρικούς εξυπηρετητές έως το Napster

Για να κατανοήσει κανείς την εξέλιξη των δικτύων ομότιμων κόμβων, καλό θα ήταν να ξεκινήσει από μια αναδρομή στην ιστορία των υπολογιστικών συστημάτων. Πριν από τη μαζική εξάπλωση των προσωπικών υπολογιστών (Personal Computers - PCs) το κόστος κτήσης και συντήρησης ενός ισχυρού υπολογιστικού συστήματος ήταν σχετικά μεγάλο. Εξ'αιτίας αυτού του περιορισμού, υπήρχαν λίγοι (σχετικά με σήμερα) ισχυροί υπολογιστές, στους οποίους αποθηκεύονταν και επεξεργάζονταν κεντρικά τα δεδομένα. Οι υπολογιστές αυτοί είχαν το ρόλο του εξυπηρετητή (server) και η πρόσβαση σε αυτούς εξυπηρετούνταν κυρίως μέσω απλών τερματικών (dumb terminals).

Καταλυτικό ρόλο στην αλλαγή της κατάστασης έπαιξε η ραγδαία εξάπλωση των προσωπικών υπολογιστών αλλά και της τεχνολογίας των τοπικών δικτύων (Local Area Networks - LANs). Σύντομα, κάθε χρήστης απέκτησε αρκετή υπολογιστική ισχύ για να εργάζεται χωρίς τη βοήθεια ενός κεντρικού εξυπηρετητή. Επιπλέον, μέσω των τοπικών δικτύων, οι χρήστες μπορούσαν να ανταλλάσσουν δεδομένα άμεσα [1], και πάλι χωρίς τη διαμεσολάβηση κάποιου server. Αν και σπάνια αναφέρεται σήμερα, αυτή ήταν ουσιαστικά η πρώτη γενιά των δικτύων ομότιμων κόμβων. Η φράση 'ομότιμοι κόμβοι' αναφέρεται ακριβώς στο γεγονός ότι όλοι οι κόμβοι του δικτύου είναι ισοδύναμοι, δηλαδή δεν υπάρχει ένας κόμβος-εξυπηρετητής και πολλοί κόμβοι-πελάτες, αλλά ο κάθε κόμβος είναι ταυτόχρονα εξυπηρετητής και πελάτης. Ο αντίστοιχος αγγλικός όρος 'peer-to-peer' (συχνά συντομεύεται σε 'p2p') αναφέρεται ομοίως στην άμεση μεταφορά δεδομένων μεταξύ δύο κόμβων (peers) χωρίς κεντρική διαμεσολάβηση.

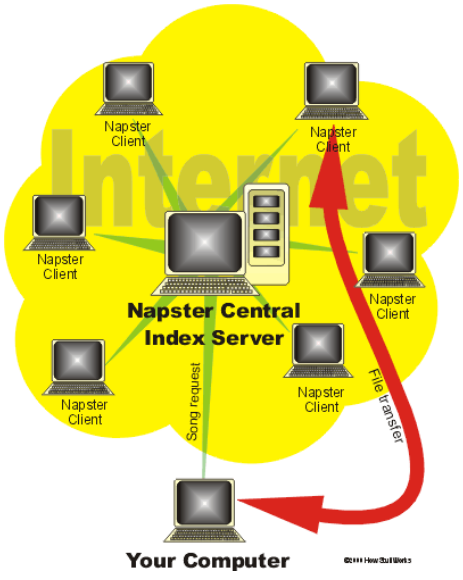
Το δεύτερο σημαντικό γεγονός που οδήγησε τα δίκτυα ομότιμων κόμβων στη σημερινή τους μορφή είναι η εκρηκτική ανάπτυξη του διαδικτύου. Μέσω του διαδικτύου, κάθε κόμβος μπορεί να συνδεθεί όχι μόνο με αυτούς που βρίσκονται στο ίδιο τοπικό δίκτυο, αλλά με εκατομμύρια άλλους κόμβους σε όλο τον κόσμο. Αυτό αύξησε σημαντικά το εύρος των

[1] με λειτουργικά συστήματα όπως Novell Netware, Windows for Workgroups κτλ.

εφαρμογών που μπορούσαν να αναπτυχθούν πάνω σε δίκτυα ομότιμων κόμβων, αλλά επέβαλλε και την ανάπτυξη νέων τεχνολογιών ώστε να υποστηριχθούν τοπολογίες που δεν περιορίζονται στα όρια ενός τοπικού δικτύου αλλά εξαπλώνονται σε όλο το διαδίκτυο.

Οι πρώτες εφαρμογές που επωφελήθηκαν από την τεχνολογία p2p ήταν οι εφαρμογές ανταλλαγής αρχείων. Παραδοσιακά, η μεταφορά αρχείων στο διαδίκτυο γινόταν από κάποιον κεντρικό εξυπηρετητή προς τον πελάτη, με χρήση των πρωτοκόλλων FTP (File Transfer Protocol - Πρωτόκολλο Μεταφοράς Αρχείων) ή HTTP (Hyper-Text Trasfer Protocol - Πρωτόκολλο Μεταφοράς Υπερκειμένου). Αυτό είχε ορισμένα μειονεκτήματα: ο κεντρικός εξυπηρετητής μπορεί να πάθει κάποια βλάβη, να υπερφορτωθεί σε περιόδους αιχμής ή απλά να βρίσκεται πολύ 'μακριά' από τον κόμβο του πελάτη με συνέπεια αργή ταχύτητα μεταφοράς. Θα δούμε παρακάτω πως η τεχνολογία των δικτύων ομότιμων κόμβων λύνει τα παραπάνω προβλήματα.

Η εφαρμογή που έστρεψε την προσοχή του κοινού στα δίκτυα p2p (killer app) ήταν το σύστημα ανταλλαγής αρχείων Napster [Tys04]. Ο τρόπος λειτουργίας του συστήματος αυτού φαίνεται στο Σχήμα 2.1. Κάθε χρήστης του συστήματος, αφού συνδεόταν σε έναν κεντρικό εξυπηρετητή, δήλωνε μια λίστα από αρχεία [2] τα οποία προτίθεται να μοιραστεί με τους άλλους χρήστες του συστήματος. Όλοι οι συνδεδεμένοι χρήστες μπορούσαν να ξεκινήσουν μια αναζήτηση για διαθέσιμα αρχεία με βάση το όνομά τους ή άλλα κριτήρια. Ο κεντρικός εξυπηρετητής επέστρεφε τότε μια λίστα από αρχεία σύμφωνα με τα κριτήρια αναζήτησης και, για κάθε αρχείο, μια λίστα των χρηστών του συστήματος που είχαν δηλώσει ότι διέθεταν το αρχείο αυτό. Εάν ο χρήστης ήθελε να μεταφέρει ένα από τα αρχεία στον υπολογιστή του, τότε συνδεόταν απευθείας με έναν ή περισσότερους από τους άλλους χρήστες.



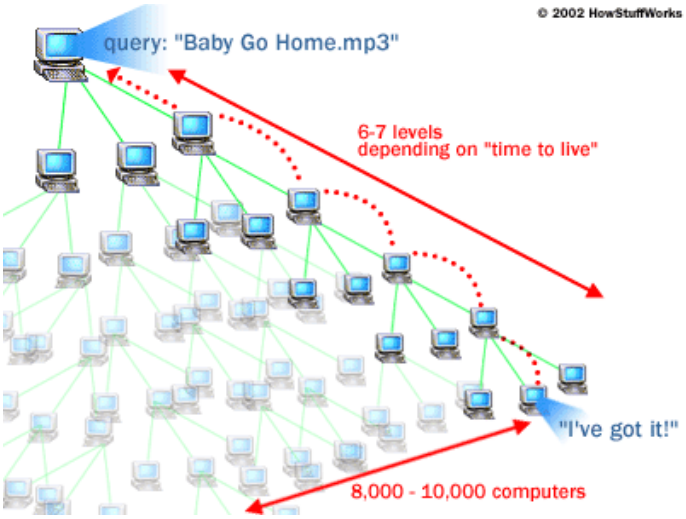
Σχήμα 2.1: Η αρχιτεκτονική του Napster (©HowStuffWorks.com)

[2] στη μεγάλη τους πλειοψηφία μουσικές ηχογραφήσεις σε μορφή MP3

Αρχιτεκτονικές δικτύων ομότιμων κόμβων

Αν και το Napster ήταν μια πρωτοπόρα εφαρμογή, στην ουσία απέχει αρκετά από το να χαρακτηριστεί μια πλήρης εφαρμογή p2p διότι, παρ'όλο που οι μεταφορές των αρχείων γίνονταν πράγματι από κόμβο σε κόμβο, η αναζήτηση των αρχείων γινόταν κεντρικά, και ο κεντρικός εξυπηρετητής ήταν ένα αναπόσπαστο τμήμα και ταυτόχρονα ο αδύναμος κρίκος του συστήματος. Μάλιστα, η λειτουργία του Napster τερματίστηκε όταν πολλές δισκογραφικές εταιρίες, ανησυχώντας για την μουσική 'πειρατεία', κέρδισαν δικαστικές μάχες με στόχο τον τερματισμό της λειτουργίας του κεντρικού εξυπηρετητή. Στις επόμενες παραγράφους θα παρουσιαστούν αρχιτεκτονικές δικτύων p2p που είναι περισσότερο αυτο-οργανούμενες και δεν εξαρτώνται από έναν κεντρικό εξυπηρετητή.

Ένα είδος αρχιτεκτονικής το οποίο υλοποιήθηκε για πρώτη φορά με το δίκτυο Gnutella [Bra04] δεν εμπεριέχει καμιά έννοια οργάνωσης και όλοι οι κόμβοι είναι πραγματικά ισότιμοι. Ένας κόμβος που θέλει να εισέλθει στο δίκτυο πρέπει να γνωρίζει τη διεύθυνση IP ενός τουλάχιστον άλλου κόμβου, ώστε να συνδεθεί με αυτόν. Με τον τρόπο αυτό δημιουργείται ένα πλέγμα (mesh) συνδεδεμένων κόμβων, όμοιο με αυτό του Σχήματος 2.2. Για την αναζήτηση ενός αρχείου, ένας κόμβος μεταβιβάζει το μήνυμα σε όλους τους γείτονές του, οι οποίοι απαντούν αν διαθέτουν το αρχείο αυτό, ενώ ταυτόχρονα προωθούν το μήνυμα στους δικούς τους γείτονες κοκ. Προφανώς υπάρχει περιορισμός στον αριθμό των κόμβων που μπορεί να διανύσει κάθε μήνυμα (ο αριθμός αυτός είναι γνωστός και ως time-to-live, TTL), ειδάλλως η ροή των μηνυμάτων θα ήταν απεριόριστη και το δίκτυο θα κατέρρεε.



Σχήμα 2.2: Η αρχιτεκτονική του Gnutella (©HowStuffWorks.com)

Η προσέγγιση αυτή έχει ένα σημαντικό πλεονέκτημα, την πλήρη αποκέντρωση του δικτύου που εξασφαλίζει πως όσο υπάρχουν κόμβοι στο δίκτυο και μπορούν να συνδεθούν μεταξύ τους, θα σχηματίσουν ένα λειτουργικό σύστημα ανταλλαγής αρχείων. Δυστυχώς όμως υπάρχει κι ένα μεγάλο πρόβλημα, ότι ο αλγόριθμος της αναζήτησης δε μπορεί να ακολουθήσει τη μεγάλη ανάπτυξη του δικτύου, και η αναζήτηση θα περιορίζεται πάντα σε μια μικρή περιοχή λόγω του μηχανισμού του TTL. Επιπλέον, έχει αποδειχθεί στην

πράξη ότι η σπατάλη των δικτυακών πόρων (protocol overhead) που προκύπτει από τη συνεχή αναμετάδοση μηνυμάτων αναζήτησης δεν είναι αμελητέα, ειδικά για κόμβους με περιορισμένης ταχύτητας πρόσβαση στο διαδίκτυο.

Είναι προφανές ότι οι δύο παραπάνω αρχιτεκτονικές αποτελούν ακραίες λύσεις για την οργάνωση ενός δικτύου ομότιμων κόμβων, γι'αυτό και στην πράξη αποδεικνύονται προβληματικές. Στην πραγματικότητα υπάρχει συνήθως μια ενδιάμεση λύση. Η λύση αυτή, που παρέχει ένα εξίσου αποκεντρωμένο αλλά ταυτόχρονα οργανωμένο δίκτυο, εφαρμόζεται πλέον στα περισσότερα σύγχρονα δίκτυα p2p, όπως το Kazaa (δίκτυο FastTrack) και το eMule (δίκτυο ED2K).

Στα δίκτυα αυτά υπάρχει μια τοπική οργάνωση, με την έννοια ότι οι κόμβοι οργανώνονται σε ομάδες και ένας κόμβος από κάθε ομάδα παίζει το ρόλο ενός τοπικού εξυπηρετητή ή αλλιώς υπερ-κόμβου (super-peer). Όλοι οι κόμβοι μιας ομάδας δηλώνουν στον τοπικό εξυπηρετητή τα αρχεία τα οποία διαθέτουν, και ο εξυπηρετητής διατηρεί μια τοπική λίστα, την οποία μάλιστα ανταλλάσσει περιοδικά και με άλλους ύπερ-κόμβους. Έτσι, κάθε ερώτημα για αναζήτηση αρχείων εξυπηρετείται αποκλειστικά από το δίκτυο των super-peers, με πιο αποτελεσματικό και επεκτάσιμο (scalable) τρόπο απ'ότι λ.χ. στο δίκτυο Gnutella. Επιπλέον, τυχόν προβλήματα σε έναν υπερ-κόμβο επηρεάζουν το δίκτυο μόνο τοπικά και προσωρινά, καθώς μπορεί είτε να επιλεγεί αυτόματα μεταξύ των μελών της ομάδας ένας νέος υπερ-κόμβος, είτε τα μέλη της ομάδας να συνδεθούν σε έναν άλλο υπερ-κόμβο και η λειτουργία να συνεχίσει κανονικά.

Ας τονιστεί στο σημείο αυτό πως όλες οι παραπάνω αρχιτεκτονικές διαφέρουν μόνο στη λογική οργάνωση του δικτύου και στον τρόπο αναζήτησης πληροφοριών. Η μεταφορά αρχείων ή άλλων δεδομένων γίνεται πάντα από έναν κόμβο απευθείας σε κάποιον άλλο και η αξιοπιστία της εξαρτάται πάντα από τη διαθεσιμότητα των επικοινωνούντων κόμβων και της μεταξύ της δικτυακής σύνδεσης. Σχετικά με τη μεταφορά δεδομένων, τα δίκτυα p2p έχουν επιπλέον ένα σημαντικό πλεονέκτημα: είναι δυνατόν να αυξηθεί η ταχύτητας λήψης των δεδομένων αλλά και να κατανεμηθεί ο δικτυακός φόρτου με παράλληλη λήψη δεδομένων από περισσότερους του ενός κόμβους. Στη συντριπτική πλειοψηφία των σύγχρονων δικτύων p2p, τα αρχεία που μεταφέρονται διαιρούνται σε τμήματα, και είναι δυνατόν ένας κόμβος να λαμβάνει παράλληλα διαφορετικά τμήματα του αρχείου από διαφορετικούς κόμβους, τα οποία στη συνέχεια τοποθετούνται στην κατάλληλη σειρά και σχηματίζουν το πρωτότυπο αρχείο.

Αναζήτηση στα δίκτυα ομότιμων κόμβων

Μέχρι στιγμής είδαμε ότι τα δίκτυα ομότιμων κόμβων μπορούν, με τη χρήση κατάλληλων αρχιτεκτονικών, να αυτο-οργανωθούν και να μεταφέρουν δεδομένα με αποτελεσματικό τρόπο. Όμως η αλματώδης ανάπτυξη των δικτύων p2p έχει φέρει στο προσκήνιο ένα πιο σημαντικό πρόβλημα: την αποτελεσματική αναζήτηση δεδομένων τα οποία είναι διασπαρμένα σε δίκτυα p2p.

Τα περισσότερα δίκτυα p2p που λειτουργούν σήμερα υποστηρίζουν αναζήτηση με πολύ

απλοϊκά κριτήρια. Στις περισσότερες περιπτώσεις χρησιμοποιούνται λέξεις κλειδιά οι οποίες πρέπει να περιέχονται στο όνομα του αρχείου. Αρκετές φορές τίθεται περιορισμός στον τύπο του αρχείου που αναζητείται, δηλαδή περιορίζεται οι αναζήτηση σε αρχεία ήχου, κειμένου, video κ.ο.κ. Η αναζήτηση αυτή γίνεται συνήθως με βάση την 'επέκταση' του αρχείου (π.χ. `.mp3`, `.pdf`, `.avi` κ.ο.κ.) αλλά η μέθοδος αυτή δε λειτουργεί πάντα αξιόπιστα: μπορεί για παράδειγμα ένα αρχείο κειμένου να περιέχεται σε ένα συμπιεσμένο σύνολο αρχείων (archive). Σπανιότατα γίνεται αναζήτηση με περισσότερο εξειδικευμένα κριτήρια, όπως το όνομα του συγγραφέα ενός κειμένου ή ο πραγματικός τίτλος[3] ενός video. Η υποστήριξη για τέτοιες εξειδικευμένες αναζητήσεις στα περισσότερα δίκτυα p2p είναι στοιχειώδης, ενώ και τα ίδια τα αρχεία πολλές φορές δεν επισημαίνονται με τα κατάλληλα μετα-δεδομένα, παρ'όλο που σε κάποιες περιπτώσεις έχει προβλεφθεί η απαραίτητη υποδομή.

Η μέθοδος αναζήτησης που περιγράψαμε στην προηγούμενη παράγραφο συνήθως επαρκεί για την απλή αναζήτηση αρχείων μουσικής και video, τα οποία απαρτίζουν και την πλειοψηφία των αρχείων που διακινούνται σήμερα σε δίκτυα p2p. Ταυτόχρονα όμως περιορίζεται η ανάπτυξη των δικτύων p2p, καθώς τα απλά αυτά κριτήρια δεν μπορούν να υποστηρίξουν αναζήτηση σε πολύπλοκο και δομημένο περιεχόμενο, π.χ. εκπαιδευτικό υλικό, επιστημονικές δημοσιεύσεις. Για να είναι αποτελεσματική η αναζήτηση σε τέτοιου είδους περιεχόμενο, λαμβάνεται συνήθως υπ'όψη ένα πλούσιο σύνολο μετα-δεδομένων.

Η νέα γενικά δικτύων p2p βασίζεται στην ενσωμάτωση ενός ολοκληρωμένου συστήματος διαχείρισης μετα-δεδομένων στην καρδιά του δικτύου p2p, με σκοπό την οριστική λύση των παραπάνω προβλημάτων. Στα δίκτυα αυτά, τα μετα-δεδομένα περιγράφονται με συγκεκριμένα σχήματα (schemas), όπου είτε όλοι οι κόμβοι έχουν το ίδιο σχήμα, είτε κάθε κόμβος έχει τη δυνατότητα να ορίσει το δικό του σχήμα, πάντα μέσα σε κάποια όρια. Με τη δεύτερη περίπτωση, η οποία είναι η πλέον ρεαλιστική και ενδιαφέρουσα, ασχολείται η παρούσα διπλωματική καθώς και οι σχετικές εργασίες που παρουσιάζονται στην Ενότητα 2.2.

2.1.2 Το πλαίσιο RDF

Ένα σύγχρονο πρότυπο περιγραφής των μετα-δεδομένων καθώς και του σχήματος που τα διέπει είναι το πλαίσιο RDF (Resource Description Framework). Το πρότυπο αυτό ήδη χρησιμοποιείται στη νέα γενιά δικτύων p2p και θα αποτελέσει και τη βάση του δικού μας συστήματος. Το RDF έχει ορισμένες ιδιότητες που αποδεικνύονται πολύ χρήσιμες σε ένα δυναμικό περιβάλλον όπως αυτό των δικτύων p2p:

- προσφέρει επεκτασιμότητα και δυνατότητα επανάχρησης σχημάτων μέσω ενός μηχανισμού κληρονομικότητας
- βοηθά την οργάνωση των σχημάτων μέσω του μηχανισμού των χώρων ονομάτων (namespaces)
- δεν επιβάλλει την πληρότητα, αλλά επιτρέπει και 'ελλιπείς' περιγραφές με βάση μόνο τα διαθέσιμα στοιχεία

[3] σε αντίθεση με τον τίτλο που έχει το αρχείο

Σύντομη περιγραφή του RDF

Το πλαίσιο RDF [MM04] ξεκίνησε ως μια προσπάθεια συστηματοποίησης του τρόπου περιγραφής μέτα-δεδομένων (metadata) που συνοδεύουν σελίδες στο Internet, για παράδειγμα πληροφορίες για το δημιουργό, τα δικαιώματα χρήσης, την καταλληλότητα για ανηλίκους κ.ο.κ. Καθώς τα μετα-δεδομένα αυτά έχουν σκοπό την υποβοήθηση της αναζήτησης και άλλων αυτοματοποιημένων λειτουργιών, είναι σημαντικό να είναι αναγνώσιμα και επεξεργάσιμα από προγράμματα (machine readable) χωρίς ανθρώπινη επέμβαση.

Για παράδειγμα, μια σελίδα μπορεί να περιέχει τη φράση `This article was written by John Doe`. Αυτό όμως δε σημαίνει τίποτα για ένα μηχάνημα. Θα μπορούσε παρόμοια να υπήρχε η φράση Το άρθρο αυτό γράφτηκε από το Γιάννη Γεωργίου ή κάτι παρόμοιο σε διαφορετική γλώσσα ή διατύπωση. Όμοια προβλήματα υπάρχουν, για παράδειγμα, σε έναν τόπο ηλεκτρονικού εμπορίου όπου η τιμή ενός προϊόντος δεν περιγράφεται με συστηματικό τρόπο ή σε έναν τόπο με ειδήσεις όπου δεν είναι εύκολο να ανακτήσει κανείς αυτόματα τις επικεφαλίδες των πρόσφατων ειδήσεων.

Με τη χρήση του πλαισίου RDF, το παραπάνω παράδειγμα θα μπορούσε να περιγραφεί συστηματικά και σε μορφή κατανοητή από μια μηχανή με το παρακάτω κείμενο, η σημασία το οποίου θα αναλυθεί στη συνέχεια:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
    <rdf:Description rdf:about="http://www.example.org/article42.html">
      <dc:creator>John Doe</dc:creator>
    </rdf:Description>
</rdf:RDF>
```

Στην ουσία κάθε περιγραφή RDF είναι ένα σύνολο από δηλώσεις. Κάθε δήλωση αποτελείται από ένα υποκείμενο (subject), ένα κατηγόρημα (predicate) και ένα αντικείμενο (object). Τα τρία αυτά μέρη κάθε δήλωσης σχηματίζουν, σύμφωνα με την ορολογία του RDF, μια τριάδα (triple).

Στο παράδειγμά μας, η δήλωση σε φυσική γλώσσα θα μπορούσε να γραφεί ως «*Το http://www.example.org/article42.html έχει έναν δημιουργό ο οποίος είναι ο John Doe*». Το ίδιο πράγμα γράφεται με τη μορφή «τριάδας» όπως παρακάτω:

Πίνακας 2.1: RDF Triple

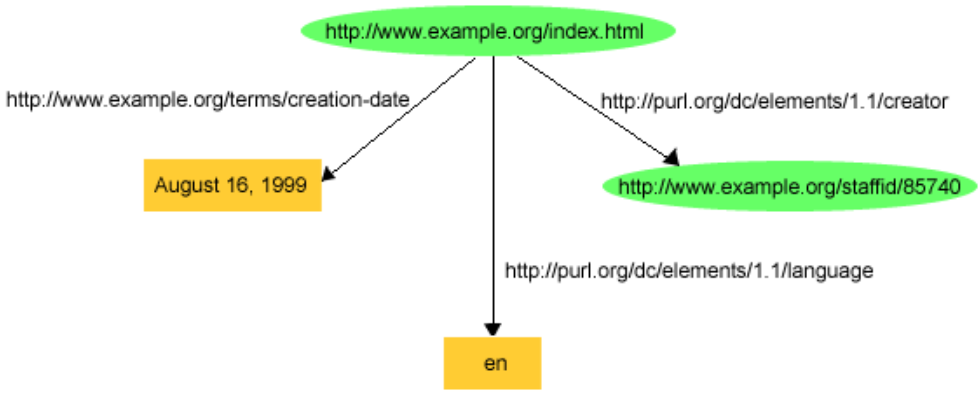
subject	predicate	object
`http://www.example.org/article42.html`	`dc:creator`	`John Doe`

Μέχρι στιγμής έχει περιγραφεί ένας τρόπος γραφής των δηλώσεων, αλλά δεν έχει ειπωθεί τίποτα για τη φύση των πραγμάτων που συμμετέχουν στις δηλώσεις αυτές. Εφ'όσον απαιτούμε οι δηλώσεις μας να είναι machine-readable, πρέπει το ίδιο να συμβαίνει και για τα πράγματα αυτά. Το RDF ξεχωρίζει δύο περιπτώσεις. Στην πρώτη περίπτωση, όταν μιλάμε

για πόρους (resources), τα πράγματα που περιγράφονται αναγνωρίζονται με τη μορφή ενός URI-Reference. Ένα URI-Reference είναι ένα URI [4] που συνοδεύεται προαιρετικά από ένα Fragment-Identifier, δηλαδή το χαρακτήρα '#' και μια συμβολοσειρά. Ένα παράδειγμα ενός έγκυρου URI-Reference που περιλαμβάνει ένα Fragment-Identifier (`#id324`) είναι το `http://www.example.org/products#id324`. Ας σημειωθεί στο σημείο αυτό ότι ένας πόρος, παρ'ότι αναγνωρίζεται μέσω ενός URI-Reference, δεν είναι απαραίτητο να είναι ένα αντικείμενο διαθέσιμο στο διαδίκτυο. Ένα URI-Reference μπορεί στην πραγματικότητα να αναφέρεται σε οποιοδήποτε αφηρημένο ή πραγματικό αντικείμενο ή έννοια. Στη δεύτερη περίπτωση, που αφορά μόνο τα αντικείμενα δηλώσεων, μιλάμε για literals τα οποία είναι μια συμβολοσειρά χαρακτήρων. Τα literals μπορούν προαιρετικά να έχουν και κάποιον τύπο (typed literals), όπου το RDF 'δανείζεται' τους τύπους που ορίζονται στην προδιαγραφή XML Schema [5].

Σύμφωνα με την τελευταία παράγραφο, στο παράδειγμά μας το υποκείμενο είναι URI-Reference και το αντικείμενο είναι literal. Αυτό που δεν είναι προφανές είναι ότι και το κατηγόρημα είναι URI-Reference. Αυτό ισχύει αν ορίσουμε ότι η συντομογραφία `dc:` είναι ισοδύναμη με `http://purl.org/dc/elements/1.1/`, οπότε το κατηγόρημα μετατρέπεται σε `http://purl.org/dc/elements/1.1/creator`.

Ένα σύνολο από δηλώσεις (statements) του RDF μπορεί να πάρει ισοδύναμα τη μορφή ενός γράφου, όπου οι κόμβοι αντιπροσωπεύουν πόρους ή literals και οι ακμές αντιπροσωπεύουν κατηγορήματα, είναι δε προσανατολισμένες από το υποκείμενο προς το αντικείμενο Ένα παράδειγμα τέτοιου γράφου φαίνεται στο σχήμα 2.3. Όταν το RDF απεικονίζεται ως γράφος, οι πόροι έχουν κατά σύμβαση οβάλ σχήμα και τα literals παραλληλόγραμμο.



Σχήμα 2.3: Ένας γράφος RDF

Μέχρι στιγμής έχουν παρουσιαστεί δυο τρόποι για την παρουσίαση δηλώσεων του RDF: η μορφή μιας λίστας από τριάδες και η μορφή γράφου. Όμως καμιά από τις μορφές αυτές δεν είναι άμεσα επεξεργάσιμη από έναν υπολογιστή. Για να εξυπηρετηθεί ο σκοπός αυτός, το RDF γράφεται σχεδόν πάντα σε μορφή XML[6]. Για παράδειγμα, ο γράφος που εικονίζεται

[4] βλ. RFC 2396 για τον πλήρη ορισμό
[5] http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#built-in-datatypes
[6] βλέπε http://www.w3.org/XML/

στο σχήμα 2.3 γράφεται ισοδύναμα σε RDF/XML ως εξής:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:exterms="http://www.example.org/terms/">

   <rdf:Description rdf:about="http://www.example.org/index.html">
      <exterms:creation-date>August 16, 1999</exterms:creation-date>
      <dc:language>en</dc:language>
      <dc:creator rdf:resource="http://www.example.org/staffid/85740"/>
   </rdf:Description>
</rdf:RDF>
```

Η ετικέτα (tag) `rdf:Description` περικλείει μια ή περισσότερες δηλώσεις για το υποκείμενο που ορίζεται στην παράμετρο `rdf:about`. Κάθε μια από τις δηλώσεις αυτές αποτελείται από μια ετικέτα με το κατηγόρημα της δήλωσης (π.χ. `dc:creator`). Το αντικείμενο της δήλωσης είτε δηλώνεται με την παράμετρο `rdf:resource`, εάν είναι πόρος, είτε περικλείεται από ετικέτα, εάν είναι literal. Παρατηρούμε ότι με τη βοήθεια του μηχανισμού XML Namespaces[7] επιτυγχάνεται η σύντομη γραφή των URI-References, όπως είχαμε περιγράψει και παραπάνω.

RDF Schema

Αν και το RDF μας δίνει έναν συστηματικό τρόπο να κάνουμε δηλώσεις, δεν βοηθά σε τίποτα στον καθορισμό του λεξιλογίου των δηλώσεων αυτών. Ένα «λεξιλόγιο» (vocabulary) είναι αναγκαίο ούτως ώστε κάποιος που θα διαβάσει ορισμένες δηλώσεις RDF να μπορεί να αποφανθεί για την εγκυρότητα και τη σημασία τους. Για παράδειγμα, μια εφαρμογή που θα επεξεργάζεται τα μετα-δεδομένα για σελίδες στο internet χρειάζεται να γνωρίζει εκ των προτέρων τι είδους δηλώσεις θα περιμένει, όπως και τι είδους μπορούν να είναι τα αντικείμενα και υποκείμενα των δηλώσεων αυτών. Αυτό μπορεί να γίνει μόνο εάν όσοι δημιουργούν και επεξεργάζονται τα συγκεκριμένα μετα-δεδομένα έχουν συμφωνήσει σε μια «κοινή γλώσσα», π.χ. καθορίζοντας π.χ. ότι για ένα *διαδικτυακό τόπο* μπορεί να οριστεί ο *δημιουργός* του δηλώνοντας το όνομά του σε μορφή *συμβολοσειράς*. Το RDF Schema δίνει έναν τρόπο για τη συστηματική περιγραφή τέτοιων δηλώσεων.

Το RDF Schema [BG04] επιτρέπει ουσιαστικά τον ορισμό δύο πραγμάτων: μιας ιεραρχίας κλάσεων (με τρόπο ανάλογο με αυτό των αντικειμενοστραφών γλωσσών προγραμματισμού) και ενός συνόλου από 'ιδιότητες' μαζί με το πεδίο ορισμού και το πεδίο τιμών καθεμιάς Είναι ενδιαφέρον, αν και μερικές φορές περιπλέκει τα πράγματα, το γεγονός ότι ορισμός των κλάσεων και των ιδιοτήτων γίνεται επίσης σε μορφή RDF, δηλαδή ως ένα σύνολο από δηλώσεις.

[7] βλέπε http://www.w3.org/TR/xml-names11/

Όσον αφορά τις κλάσεις, το RDF/S παρέχει δύο είδη δηλώσεων: αυτό του ορισμού μιας κλάσης (δηλαδή «υπάρχει μια κλάση η οποία ονομάζεται `xyz`») και αυτό του ορισμού της υπερκλάσης (δηλαδή «η κλάση `abc` είναι υποκλάση της κλάσης `xyz`»). Παρακάτω δίνεται ένα παράδειγμα γραμμένο στην απλούστερη δυνατή μορφή RDF/XML στο οποίο ορίζεται μια κλάση και μια υποκλάση της:

```
<rdfs:Class rdf:ID="Fruit"/>

<rdfs:Class rdf:ID="Apple">
  <rdfs:subClassOf rdf:resource="#Fruit"/>
</rdfs:Class>
```

Με ένα σύνολο τέτοιων δηλώσεων ορίζεται μια ιεραρχία κλάσεων όπως στις περισσότερες αντικειμενοστραφείς γλώσσες. Για να έχει σημασία η ιεραρχία αυτή, θα πρέπει να σχηματίζει έναν κατευθυνόμενο ακυκλικό γράφο (DAG).

Όσον αφορά τώρα τις ιδιότητες, είναι σημαντικό ότι στο RDF/S ορίζονται ως πρώτης τάξης (first-class) αντικείμενα όπως και οι κλάσεις. Η δήλωσή μιας ιεραρχίας ιδιοτήτων γίνεται με ανάλογο τρόπο, αντικαθιστώντας το `rdfs:Class` με `rdf:Property` και το `rdfs:subClassOf` με `rdfs:subPropertyOf`. Το νόημα μιας υπο-ιδιότητας είναι ότι αποτελεί εξειδίκευση μιας άλλης ιδιότητας, π.χ. μια ιδιότητα 'διεύθυνση' θα μπορούσε να έχει υπο-ιδιότητες 'διεύθυνση κατοικίας', 'διεύθυνση εργασίας' κ.ο.κ. ώστε οι δηλώσεις να αποκτούν μεγαλύτερη λεπτομέρεια με τη χρήση των υπο-ιδιοτήτων.

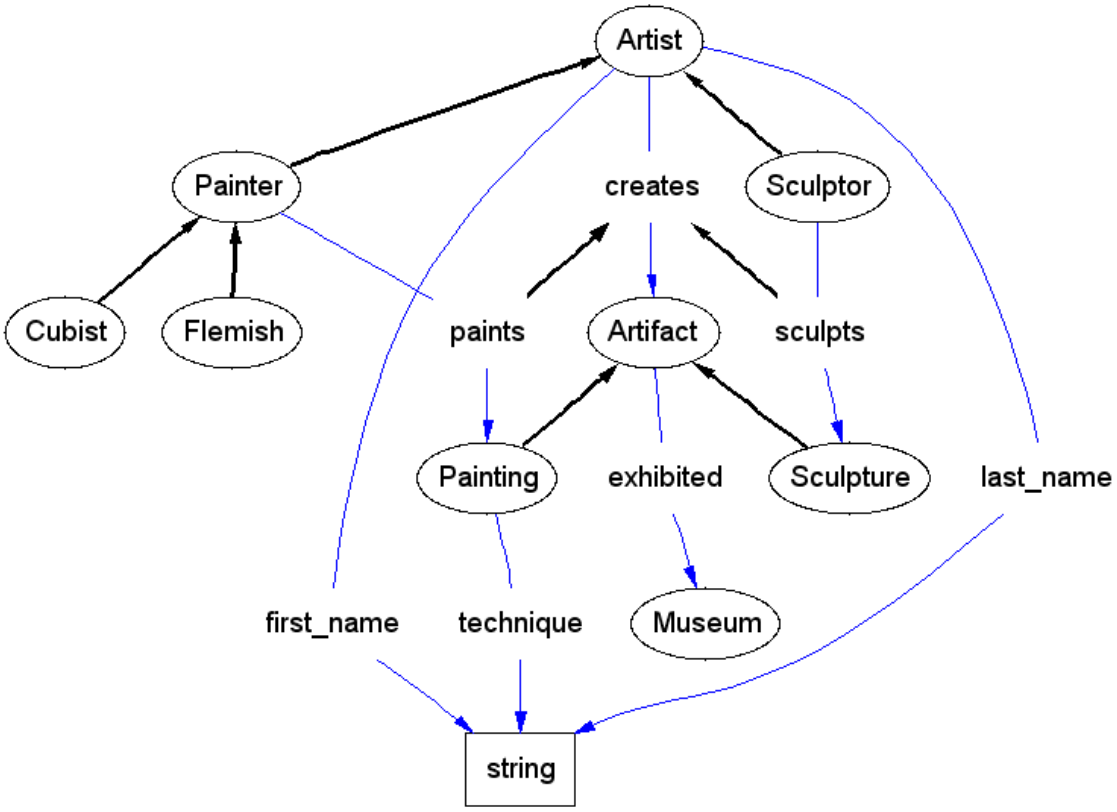
Για μια ιδιότητα μπορεί να οριστεί επίσης το πεδίο τιμών της και το πεδίο ορισμού της. Το πεδίο τιμών ορίζεται με τη δήλωση `rdfs:range` και μπορεί να είναι είτε μια κλάση είτε ένα λεκτικό με συγκεκριμένο τύπο. Το πεδίο ορισμού ορίζεται με τη δήλωση `rdfs:domain` και πρέπει να είναι μια κλάση. Σύμφωνα με τις δηλώσεις αυτές, μια ιδιότητα μπορεί να αναφέρεται μόνο σε αντικείμενα τα οποία ανήκουν στη κλάση - πεδίο ορισμού της ή σε μια από υποκλάσεις της κλάσης αυτής. Ομοίως, εάν το πεδίο τιμών της είναι μια κλάση, η ιδιότητα πρέπει να παίρνει τιμές αντικείμενα που είναι μέλη της κλάσης-πεδιού τιμών και των υποκλάσεών της. Οι υπο-ιδιότητες, τέλος, κληρονομούν το πεδίο ορισμού και το πεδίο τιμών της υπερ-ιδιότητάς τους.

Πρέπει να επισημανθούν στο σημείο αυτό ορισμένες σημαντικές διαφορές που έχει το RDF/S από τα συστήματα τύπων που ορίζονται σε γλώσσες προγραμματισμού όπως η C++ και η Java:

- Εάν δεν οριστεί πεδίο ορισμού και πεδίο τιμών για κάποια ιδιότητα, τότε η ιδιότητα μπορεί να αναφέρεται σε οποιοδήποτε αντικείμενο και να παίρνει οποιαδήποτε τιμή. Δεν είναι απαραίτητο δηλαδή κάθε ιδιότητα να ανήκει σε μια συγκεκριμένη κλάση.

- Εάν για μια ιδιότητα δηλωθούν παραπάνω από μια κλάσεις ως πεδίο ορισμού, τότε δεν ισχύει ότι η ιδιότητα αυτή μπορεί να αναφέρεται σε παραπάνω από μια κλάσεις, αλλά ότι τα αντικείμενα στα οποία αναφέρεται η ιδιότητα αυτή πρέπει να είναι ταυτόχρονα μέλη όλων αυτών τον κλάσεων.

- Δε μπορεί να ορισθεί για μια ιδιότητα διαφορετικό πεδίο τιμών ανάλογα με την κλάση στην οποία αναφέρεται.
- Όταν ορίζεται ένας πόρος που είναι στιγμιότυπο (instance) μιας κλάσης, δεν είναι υποχρεωτικό να έχει καμία από της ιδιότητες που έχουν ως πεδίο ορισμού την κλάση αυτή.



Σχήμα 2.4: Ένας γράφος RDF/S

Τα σχήματα που περιγράφονται με το RDF/S μπορούν να εμφανιστούν και γραφικά, όπως στο σχήμα 2.4. Τα οβάλ αναπαριστούν τις κλάσεις και τα παραλληλόγραμμα τους τύπους των literals. Τα βέλη με παχιά γραμμή δηλώνουν την ιεραρχία των κλάσεων και ιδιοτήτων και έχουν κατεύθυνση από την υποκλάση προς την υπερκλάση και από την υπο-ιδιότητα προς την ύπερ-ιδιότητα. Τα ονοματισμένα βέλη με λεπτή γραμμή αναπαριστούν ιδιότητες και έχουν κατεύθυνση από το πεδίο ορισμού προς το πεδίο τιμών της ιδιότητας.

Ένας τυπικός ορισμός του RDF/S ως γράφου θα δοθεί στην παράγραφο 3.1.1.

Η γλώσσα RQL

Αν και το RDF γράφεται συνήθως σε μορφή κειμένου (XML), η μορφή αυτή γενικά δεν επιτρέπει τη εύκολη διαχείριση και επεξεργασία των δεδομένων RDF και των σχημάτων RDF/S. Ταυτόχρονα, η αναζήτηση ανάμεσα σε δεδομένα RDF που έχουν τη μορφή αυτή

θα είναι, μοιραία, αρκετά αργή. Το σύστημα RDFSuite [ACK+01] παρέχει ένα σύνολο εργαλείων που επιτρέπει τη διαχείριση RDF δεδομένων και σχημάτων με χρήση μιας βάσης δεδομένων SQL, η οποία αποδεδειγμένα παρέχει υψηλή ταχύτητα. Με χρήση ενός ειδικού εργαλείου, η ΒΔ διαμορφώνεται ανάλογα με το σχήμα RDF και εισάγονται τα δεδομένα RDF. Μετά την προεργασία αυτή, παρέχονται κατάλληλα εργαλεία για την ενημέρωση των δεδομένων, αλλά κυρίως η γλώσσα RQL (RDF Query Language) για την αποτελεσματική εκτέλεση ερωτήσεων (queries) πάνω στο σχήμα και τα δεδομένα RDF.

Η πλήρης περιγραφή της γλώσσας RQL ξεφεύγει από το σκοπό της εισαγωγής αυτής, θα παρουσιάσουμε ωστόσο συνοπτικά τις βασικές δυνατότητες της γλώσσας και κυρίως αυτές που αφορούν στις ερωτήσεις πάνω σε σχήματα RDF. Οι πλήρεις οδηγίες χρήσης της γλώσσας RQL είναι διαθέσιμες στο διαδικτυακό τόπο του συστήματος RDFSuite[8].

Όσον αφορά τις ερωτήσεις πάνω στο σχήμα RDF, αυτές αφορούν την ιεραρχία των κλάσεων και των ιδιοτήτων καθώς και τα πεδία τιμών και ορισμού:

- Με τις συναρτήσεις `subClassOf(c)` και `superClassOf(c)` μπορούμε να βρούμε τις υποκλάσεις και τις υπερκλάσεις της κλάσης `c`, εφ'όσον υπάρχουν. Ομοίως ορίζονται για την ιεραρχία των ιδιοτήτων οι συναρτήσεις `subPropertyOf(p)` και `superPropertyOf(p)`. Οι συναρτήσεις αυτές βρίσκουν όλες τις υπό/ύπερκλάσεις, άμεσες ή μη. Εάν θέλουμε να περιορίσουμε την αναζήτηση μόνο στις άμεσες σχέσεις κληρονομικότητας, πρέπει να τοποθετηθεί ο χαρακτήρας ^ ανάμεσα στο όνομα και την παράμετρο της συνάρτησης, δηλαδή η `subClassOf(c)` γίνεται `subClassOf^(c)`.

- Με τις συναρτήσεις `leafclass(c)` και `leafproperty(p)` βρίσκουμε όλους τους τελικούς απογόνους της κλάσης `c` ή της ιδιότητας `p` αντίστοιχα, δηλαδή τους απογόνους που δεν έχουν άλλους απογόνους. Εάν παραλείψουμε την παράμετρο στις συναρτήσεις αυτές, θα πάρουμε όλες τις τελικές κλάσεις ή ιδιότητες του σχήματος. Ομοίως, με τη χρήση των συναρτήσεων `topclass` και `topproperty` (χωρίς παράμετρο) θα πάρουμε όλες τις κλάσεις ή ιδιότητες του σχήματος που δεν έχουν κανένα πρόγονο, βρίσκονται δηλαδή στη ρίζα κάθε ιεραρχίας.

- Η συνάρτηση `nca(a,b)` επιστρέφει τον πλησιέστερο κοινό πρόγονο των κλάσεων ή ιδιοτήτων `a,b`, εφ'όσον αυτός υφίσταται.

- Οι συναρτήσεις `domain(p)` και `range(p)` επιστρέφουν το πεδίο ορισμού και το πεδίο τιμών της ιδιότητας `p`, εφ'όσον αυτά έχουν οριστεί στο σχήμα.

Στη συνέχεια, και με τη βοήθεια κάποιων παραδειγμάτων, θα δείξουμε τη χρήση της εντολής `SELECT` της RQL:

Παράδειγμα 2.1

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

[8] http://139.91.183.30:9090/RDF/RQL/Manual.html

Υποθέτοντας ότι στο σχήμα υπάρχει μια ιδιότητα `creates`, το ερώτημα αυτό θα επιστρέψει όλα τα ζεύγη κλάσεων `C1`, `C2` όπου η `C1` ανήκει στο πεδίο ορισμού και η `C2` στο πεδίο τιμών της ιδιότητας `creates`. Οι 'μεταβλητές' του ερωτήματος που έχουν πρόθεμα το χαρακτήρα `$` θεωρούνται ότι είναι κλάσεις.

Παράδειγμα 2.2

```
SELECT @P, range(@P)
FROM   {;Painter}@P
```

Υποθέτοντας ότι στο σχήμα υπάρχει μια κλάση `Painter`, το ερώτημα αυτό θα επιστρέψει όλες τις ιδιότητες `P`, που έχουν ως πεδίο ορισμού την κλάση αυτή, καθώς και το πεδίο τιμών της καθεμιάς Οι 'μεταβλητές' του ερωτήματος που έχουν πρόθεμα τον χαρακτήρα `@` θεωρούνται ότι είναι ιδιότητες.

Παράδειγμα 2.3

```
SELECT @P, $Y
FROM   {;Painter}@P{$Y}
```

Στο παράδειγμα αυτό ζητάμε ότι και στο προηγούμενο, με τον περιορισμό όμως η ιδιότητα `P` να έχει πεδίο ορισμού μια κλάση (όχι, δηλαδή, literal). Αυτό δηλώνεται από το πρόθεμα `$` στη μεταβλητή `Y`.

Παράδειγμα 2.4

```
SELECT $X, $Z
FROM   creates{$X}, {$Y}exhibited{$Z}
WHERE  $X = $Y
```

Βλέπουμε, τέλος, ένα πιο πολύπλοκο παράδειγμα, όπου ζητούνται κλάσεις που συμμετέχουν σε παραπάνω από μια ιδιότητες και με τη χρήση της εντολής `WHERE` θέτουμε περιορισμούς ανάλογους με το join στις σχεσιακές βάσεις δεδομένων. Συγκεκριμένα, ζητούνται τα ζεύγη κλάσεων `X`, `Z` όπου η `X` ανήκει στο πεδίο τιμών της ιδιότητας `creates` και ταυτόχρονα στο πεδίο ορισμού της ιδιότητας `exhibited`, ενώ η `Z` ανήκει στο πεδίο τιμών της ιδιότητας `exhibited`. Το ίδιο παράδειγμα γράφεται ισοδύναμα:

Παράδειγμα 2.5

```
SELECT $X, $Z
FROM   creates{$X}.{$Y}exhibited{$Z}
```

Με την παραπάνω γραφή, η τελεία ανάμεσα στο {$X} και στο {$Y} εκφράζει την ισότητα (join) {$X = $Y}.

Όπως προείπαμε, σε RQL μπορούμε να γράψουμε ερωτήσεις όχι μόνο επί του σχήματος αλλά και επί των δεδομένων RDF που έχουν εισαχθεί σε μια ΒΔ σύμφωνα με το αντίστοιχο σχήμα. Δίδοντας ως ερώτημα το όνομα μιας κλάσης, π.χ. `Worker`, θα λάβουμε όλα τα

στιγμιότυπα της κλάσης αυτής και των υποκλάσεών της. Αν θέλουμε να αποκλείσουμε τις υπερκλάσεις, δηλαδή να πάρουμε μόνο τα άμεσα στιγμιότυπα της κλάσης, θα προσθέσουμε το πρόθεμα ^, π.χ. `^Worker`. Ομοίως, δίνοντας ως ερώτημα το όνομα μιας ιδιότητας, π.χ. `creates`, θα πάρουμε ζεύγη από πόρους (δηλαδή στιγμιότυπα κλάσεων) που συνδέονται με την ιδιότητα `creates`.

Τέλος, πιο πολύπλοκες ερωτήσεις πάνω στα δεδομένα μπορούν να εκφραστούν και πάλι με χρήση της εντολής `SELECT` όπως στα παρακάτω παραδείγματα:

Παράδειγμα 2.6

```
SELECT Y
FROM   {X}creationDate{Y}
WHERE  X = &http://example.org/rdfDemo
```

Στο παράδειγμα αυτό ζητείται η τιμή `Y` της ιδιότητας `creationDate` για τον πόρο ο οποίος ορίζεται με το URI `http://example.org/rdfDemo`.

Παράδειγμα 2.7

```
SELECT X, Y
FROM   {X;TruckDriver}worksFor{Y;TransportCompany}
```

Στο παράδειγμα αυτό ζητούνται και πάλι ζεύγη πόρων που συνδέονται με την ιδιότητα `worksFor`, τίθεται όμως ο περιορισμός το πρώτο μέλος κάθε ζεύγους να είναι στιγμιότυπο της κλάσης `TruckDriver` και το δεύτερο μέλος στιγμιότυπο της κλάσης `TransportCompany`.

2.2 Σχετικές εργασίες

2.2.1 Piazza

Το σύστημα Piazza [HIMT03] προσφέρει μια υποδομή διαχείρισης δεδομένων για τον σημασιολογικό ιστό (semantic web). Πρόκειται για ένα δίκτυο κόμβων οι οποίοι συνεργάζονται παρέχοντας ο καθένας τις δικές του πληροφορίες, είτε με τη μορφή δεδομένων είτε με τη μορφή σχημάτων. Το σύστημα υποστηρίζει διάφορες μορφές για τα δεδομένα και τα σχήματα: RDF σε συνδυασμό με κάποια γλώσσα περιγραφής οντολογιών (DAML+OIL, OWL) καθώς και απλά δεδομένα XML χωρίς σχήμα. Ειδικά η δεύτερη περίπτωση, παρ᾽όλο που είναι πιο 'φτωχή' εκφραστικά σε σχέση με το RDF, χρησιμοποιείται συχνότερα στο Internet καθώς πολλά σύγχρονα συστήματα λογισμικού και διαδικτυακές υπηρεσίες μπορούν άμεσα να εξάγουν δεδομένα σε κάποια μορφή XML.

Τελικός σκοπός του συστήματος είναι η ενοποίηση πολλών πηγών δεδομένων και η δυνατότητα πολύπλοκων ερωτήσεων με χρήση πολλών από τις πηγές που ανήκουν στο δίκτυο. Στην επίτευξη του στόχου αυτού συνεισφέρουν τα παρακάτω βασικά στοιχεία του συστήματος Piazza:

- Μια γλώσσα ορισμού αντιστοιχιών (mappings) μεταξύ των σχημάτων που χρησιμοποιεί ένα ζεύγος κόμβων. Το Piazza δεν απαιτεί να ακολουθούν όλοι οι κόμβοι

ένα συγκεκριμένο σχήμα, καθώς αυτό αποδεικνύεται πρακτικά αδύνατον. Αντίθετα, επικεντρώνεται στον ορισμό τοπικών αντιστοιχιών ανά ζεύγος κόμβων, κάτι το οποίο είναι συνήθως εφικτό. Με τον τρόπο αυτό, για να συνδεθεί ένας καινούριος κόμβος στο δίκτυο, αρκεί το σχήμα των δεδομένων τού να μπορεί να αντιστοιχιστεί με το σχήμα ενός τουλάχιστον άλλου κόμβου. Σημαντικό είναι επίσης το γεγονός ότι υποστηρίζονται αντιστοιχίσεις τόσο μεταξύ ζευγών με όμοια μορφή δεδομένων (δηλ. RDF-RDF και XML-XML) όσο και μεταξύ ζευγών με ανόμοια μορφή, δηλαδή αντιστοίχηση RDF με XML.

- Ένας αλγόριθμος απάντησης ερωτημάτων, ο οποίος λαμβάνει υπ'όψιν τους παραπάνω μετασχηματισμούς. Ο αλγόριθμος αυτός μετατρέπει το αρχικό ερώτημα σε ένα άλλο ερώτημα ή σύνολο ερωτημάτων που μπορούν να απαντηθούν με βάση τα επιμέρους σχήματα κάθε κόμβου (query rewriting).

Ένα παράδειγμα που παρουσιάζουν οι δημιουργοί του συστήματος Piazza είναι η διασύνδεση πολλών πηγών πληροφοριών σχετικά με την έρευνα στο πεδίο των βάσεων δεδομένων. Οι πηγές περιλαμβάνουν, για παράδειγμα, πανεπιστημιακές ερευνητικές ομάδες, βάσεις δεδομένων για ερευνητικές εργασίες και γνωστά συνέδρια στο χώρο των ΒΔ. Στόχος είναι η διατύπωση ερωτήσεων προς αυτό το σώμα πληροφοριών με ενιαίο τρόπο, παρ'όλη τη διαφορετική μορφή (και το διαφορετικό πεδίο) των δεδομένων που παρέχεται από κάθε πηγή.

2.2.2 Edutella

Το σύστημα Edutella [NWQ+02] στοχεύει στη λύση του προβλήματος της αναζήτησης σε δίκτυα ομότιμων κόμβων με χρήση πλούσιων μετα-δεδομένων. Αρχικό πεδίο εφαρμογής του Edutella είναι το εκπαιδευτικό υλικό, δηλαδή βιβλία, σημειώσεις διαλέξεων, διαδικτυακοί τόποι εκμάθησης εξ'αποστάσεως κ.α., μια και το υλικό είναι πολύ χρήσιμο να περιγράφεται από αναλυτικά μετα-δεδομένα, για τα οποία μάλιστα έχουν αναπτυχθεί εξειδικευμένα πρότυπα (IEEE LOM, ADL SCORM κ.α.).

Το Edutella βασίζεται στο πλαίσιο RDF, το οποίο είδαμε ότι προσφέρει μια ευέλικτη λύση για την αναπαράσταση μεταδεδομένων, καθώς και στην τεχνολογία JXTA για την ανάπτυξη και διαχείριση p2p δικτύων. Οι βασικές υπηρεσίες που προβλέπονται για το ολοκληρωμένο σύστημα είναι:

- Υπηρεσία ερωτημάτων, η οποία παρέχει μια τυποποιημένη διεπαφή αναζήτησης και ανάκτησης RDF δεδομένων
- Υπηρεσία αναπαραγωγής (replication), η οποία διαχειρίζεται την κατανομή των δεδομένων στους κόμβους του δικτύου με σκοπό την ασφάλεια των δεδομένων (redundancy) και την κατανομή του φόρτου εργασίας
- Υπηρεσία αντιστοιχίσεων, η οποία διαχειρίζεται τις αντιστοιχίσεις (mappings) μεταξύ κόμβων με διαφορετικά σχήματα

- Υπηρεσία διαμεσολάβησης (mediation), η οποία συγκεντρώνει και συγχωνεύει δεδομένα από πολλούς κόμβους
- Υπηρεσία επισήμανσης (annotation), η οποία προσφέρει τη δυνατότητα επεξεργασίας των μέτα-δεδομένων του υλικού που αποθηκεύεται στο δίκτυο.

Για την ανταλλαγή ερωτημάτων και δεδομένων μεταξύ των κόμβων, το Edutella ορίζει τη δικιά του γλώσσα ερωτημάτων RDF-QEL (RDF Query Exchange Language) καθώς και ένα πρότυπο μοντέλο δεδομένων ECDM (Edutella Common Data and Query Exchange Model). Για τη διατύπωση ερωτημάτων από τους χρήστες υποστηρίζεται επίσης μια γραφική διεπαφή χρήστη καθώς και διάφορες άλλες γλώσσες ερωτημάτων, όπως η RQL, η TRIPLE κ.α.

Η απάντηση των ερωτημάτων στο Edutella γίνεται μέσω της υπηρεσίας διαμεσολάβησης και των αντίστοιχων διαμεσολαβητών κόμβων (meditator peer). Ένας απλός διαμεσολαβητής μπορεί μόνο να προωθεί το ερώτημα σε ένα κόμβο που έχει δηλώσει ότι διαθέτει τα απαραίτητα δεδομένα. Ένας πιο πολύπλοκος διαμεσολαβητής συλλέγει δεδομένα από παραπάνω από έναν κόμβους και τα μετατρέπει σε ομοιογενή μορφή με βάση τις αντιστοιχίσεις που έχουν δηλωθεί.

Το λογισμικό του δικτύου Edutella περιλαμβάνει ήδη διάφορους τύπους έτοιμων κόμβων, οι οποίοι συνεργάζονται μεταξύ τους χρησιμοποιώντας τη γλώσσα RDF-QEL και το μοντέλο ECDM. Κάποιοι από τους κόμβους αυτούς αναλαμβάνουν την αποθήκευση των μέτα-δεδομένων με διάφορους τρόπους και σύμφωνα με διαφορετικά πρότυπα. Κάποιοι άλλοι χρησιμοποιούνται για τη διατύπωση ερωτημάτων σε κάποια από τις γλώσσες που αναφέραμε παραπάνω. Τέλος, υπάρχουν οι κόμβοι που αναλαμβάνουν τη διαμεσολάβηση και απάντηση ερωτημάτων.

2.2.3 SQPeer

Το σύστημα SQPeer [KC04] είναι ένα σύστημα σημασιολογικής δρομολόγησης και απάντησης ερωτημάτων (Semantic Query Routing and Processing) για δίκτυα p2p. Σε ένα δίκτυο p2p όπου υλοποιείται η προσέγγιση του SQPeer θεωρούμε ότι έχουμε πολλούς κόμβους, οι οποίοι διαθέτουν σχήματα και δεδομένα σε μορφή RDF. Ο καθένας από τους κόμβους αυτό μπορεί να έχει διαφορετικό 'ενεργά' σχήματα. Ενεργό θεωρείται το υποσύνολο του σχήματος ενός κόμβου για το οποίο υπάρχουν αποθηκευμένα δεδομένα. Για παράδειγμα μια κλάση C η οποία ανήκει στο αρχικό σχήμα ενός κόμβου, αλλά στα δεδομένα του ίδιου κόμβου δεν βρίσκεται κανένα στιγμιότυπό της, δεν θα συμπεριληφθεί στο ενεργό σχήμα. Με τον τρόπο αυτό εξασφαλίζεται ότι, αν ένας κόμβος λάβει ένα ερώτημα που αφορά μόνο το ενεργό του σχήμα, θα είναι σε θέση να επιστρέψει κάποια δεδομένα.

Η γλώσσα που χρησιμοποιείται για τα ερωτήματα είναι η RQL, και τα ερωτήματα τα οποία επωφελούνται από τους αλγορίθμους του SQPeer είναι κυρίως ερωτήματα που περιέχουν εκφράσεις τύπου join, δηλαδή διάσχιση ενός γράφου RDF όπως περιγράφεται στα Παραδείγματα 2.4 και 2.5 της Ενότητας 2.1.2. Για κάθε ερώτημα προς εκτέλεση κατασκευάζεται

ένας γράφος (query pattern graph) ο οποίος προκύπτει από τη δήλωση `FROM` της RQL και αντιπροσωπεύει το τμήμα του σχήματος (δηλαδή τις κλάσεις και τις ιδιότητες που τις συνδέουν) το οποίο συμμετέχει στην απάντηση του ερωτήματος.

Στη συνέχεια, ο αλγόριθμος δρομολόγησης ερωτημάτων (Query-Routing Algorithm) αναζητά ομοιότητες ανάμεσα στο γράφο του ερωτήματος και στα ενεργά σχήματα κάθε κόμβου. Κάθε τμήμα του γράφου του ερωτήματος το οποίο περιέχεται στο ενεργό σχήμα κάποιου κόμβου επισημαίνεται με το όνομα του κόμβου αυτού. Με τον τρόπο αυτό βρίσκονται για κάθε ερώτημα όλοι οι κόμβοι οι οποίοι διαθέτουν δεδομένα σχετικά με το προς εκτέλεση ερώτημα. Τέλος, ο αλγόριθμος απάντησης ερωτημάτων (Query-Processing Algorithm) διαβάζει τον γράφο του ερωτήματος επισημασμένο με τα ονόματα των κόμβων και παράγει το σχέδιο εκτέλεσης (query plan) του ερωτήματος, δηλαδή το ερώτημα που θα εκτελέσει κάθε κόμβος ανάλογα με το ενεργό σχήμα του και την πορεία συλλογής και συγχώνευσης των αποτελεσμάτων.

Ο αλγόριθμος απάντησης των ερωτημάτων μπορεί να λάβει υπ'όψιν του επιπλέον πληροφορίες, εφ*όσον αυτές είναι διαθέσιμες. Τέτοιες πληροφορίες μπορεί να είναι:

- Γνωστές αντιστοιχίες (mappings) μεταξύ σχημάτων δύο κόμβων, για να γίνουν οι κατάλληλες μετατροπές σε ερωτήματα που στέλνονται σε κόμβους με διαφορετικά σχήματα.

- Πληροφορίες για τον υπολογιστικό φόρτο των κόμβων και της δικτυακής σύνδεσής τους, έτσι ώστε να επιλεγεί ο λιγότερο απασχολημένος κόμβος μεταξύ ενός συνόλου κόμβων που αποθηκεύουν τα ίδια δεδομένα.

2.3 Στόχος

Όπως είδαμε παραπάνω, τα δίκτυα ομότιμων κόμβων προσφέρουν έναν αποτελεσματικό τρόπο κατανεμημένης αποθήκευσης και διαχείρισης της πληροφορίας, ελαχιστοποιώντας το κόστος οργάνωσης του δικτύου και τις ζημιές σε περίπτωση αστοχίας σε ένα ή περισσότερα σημεία του. Από την άλλη πλευρά, το πλαίσιο RDF και οι σχετικές με αυτό τεχνολογίες (RDF Schema, RQL) παρέχουν έναν ευέλικτο και αρκετά εκφραστικό τρόπο φορητής αναπαράστασης κάθε είδους πληροφορίας, καθώς και τη δυνατότητα σύνθετων ερωτήσεων πάνω στην πληροφορία αυτή.

Στόχος της παρούσας εργασίας είναι να διερευνηθεί η ενοποίηση των τεχνολογιών ομότιμων δικτύων και RDF, μέσω της μελέτης, σχεδίασης και ανάπτυξης ενός κατανεμημένου συστήματος αποθήκευσης πληροφορίας σε ένα σύνολο κόμβων. Συγκεκριμένα, θα αναπτυχθούν τα παρακάτω:

- Ένα δίκτυο p2p με σχήματα RDF (RDF Schema-Based p2p). Κάθε κόμβος του δικτύου αυτού θα μπορεί να αποθηκεύει RDF σχήματα και δεδομένα καθώς και να απαντά σε ερωτήματα RQL με βάση τα αποθηκευμένα σχήματα και δεδομένα.

- Ένας ειδικός κόμβος του δικτύου αυτού ο οποίος θα έχει επιπλέον δυνατότητες σχετικές με την οργάνωση και επίβλεψη του δικτύου (supervisor peer)
- Μια γεννήτρια τυχαίων τοπολογιών δικτύου, με βάση τα αποτελέσματα της οποίας θα αναπτύσσεται το δίκτυο
- Μια γεννήτρια τυχαίων σχημάτων RDF/S τα οποία βασίζονται σε ένα αρχικό σχήμα. Τα σχήματα αυτά θα διανέμονται από τον επιβλέποντα κόμβο σε καθέναν από τους υπόλοιπους κόμβους.
- Μια γεννήτρια τυχαίων ερωτημάτων RQL, τα οποία θα στέλνονται στους κόμβους του δικτύου με σκοπό να γίνουν δοκιμές και μετρήσεις.

Τα παραπάνω στοιχεία δημιουργούν μια πλήρη πλατφόρμα πάνω στην οποία θα είναι δυνατόν να γίνουν μετρήσεις (benchmarks), δοκιμές (simulations) και γενικά πειραματισμός με διαφορετικούς αλγορίθμους απάντησης ερωτημάτων και γενικά αλγορίθμους που αφορούν p2p δίκτυα βασισμένα σε σχήματα RDF. Στοχεύουμε επίσης τα παραπάνω στοιχεία της εφαρμογής να αποτελέσουν παραδείγματα καλής χρήσης (best practice) των αντίστοιχων τεχνολογικών και να είναι όσο το δυνατόν επαναχρησιμοποιήσιμα και επεκτάσιμα.

Κεφάλαιο 3

Θεωρητική μελέτη

Όπως αναφέρθηκε και στην παράγραφο 1.1, σκοπός μας είναι στο πρότυπο σύστημα που θα αναπτύξουμε να επιτρέψουμε την ύπαρξη κόμβων με διαφορετικό τρόπο αναπαράστασης δεδομένων, δηλαδή με διαφορετικά RDF σχήματα, τα οποία όμως πρέπει να είναι 'συμβατά' μεταξύ τους. Στο κεφάλαιο αυτό θα ορίσουμε τυπικά το RDF σχήμα ως ένα γράφο καθώς και την έννοια της 'συμβατότητας' μεταξύ δυο σχημάτων RDF. Θα περιγράψουμε επίσης έναν αλγόριθμο ο οποίος, δοθέντος ενός αρχικού RDF σχήματος, μπορεί να παράγει ένα σύνολο τροποποιημένων σχημάτων συμβατών μεταξύ τους σύμφωνα με τον αντίστοιχο ορισμό.

3.1 Τυπικός ορισμός του RDF/S

3.1.1 Ορισμός και αναπαράσταση του RDF/S ως γράφου

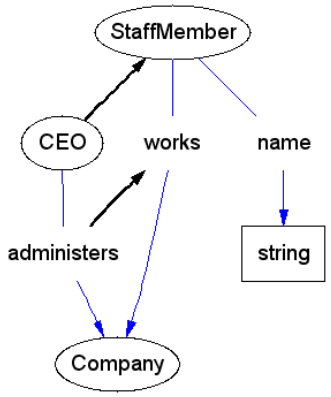
Όπως είδαμε στο προηγούμενο κεφάλαιο, τα σχήματα RDF/S μπορούν να να αναπαρασταθούν με τη μορφή ενός γράφου. Κάθε κλάση του RDF αναπαρίσταται ως ένας κόμβος και κάθε ιδιότητα ως κατευθυνόμενη ακμή. Οι ακμές έχουν κατεύθυνση από το πεδίο ορισμού προς το πεδίο τιμών της ιδιότητας, και φέρουν ως ετικέτα το όνομα της ιδιότητας.

Ορισμός 3.1 *Ένα σχήμα RDF/S είναι ένας κατευθυνόμενος γράφος (C, L, P), όπου:*

1. *C είναι ένα σύνολο ονοματισμένων κόμβων. Κάθε κόμβος στο C αντιπροσωπεύει μια κλάση του σχήματος RDF/S.*
2. *L είναι ένα σύνολο κόμβων ονοματισμένων με τύπους δεδομένων όπως* `integer`*,* `string`*, κτλ. Κάθε κόμβος στο L αντιπροσωπεύει ένα literal του σχήματος RDF/S.*
3. *P είναι ένα σύνολο ονοματισμένων ακμών (c_1, c_2, p) με κατεύθυνση από τον κόμβο c_1 στον κόμβο c_2 και όνομα p, όπου $c_1 \in C$ και $c_2 \in C \cup L$. Κάθε ακμή στο P αντιπροσωπεύει μια ιδιότητα του σχήματος RDF/S όπου $domain(p) = c_1$ και $range(p) = c_2$.*
4. *$(C, \prec)$ είναι μια σχέση μερικής διάταξης που ορίζεται για τους κόμβους του C και αναπαριστά την ιεραρχία κλάσεων ενός σχήματος RDF/S. Αν $c_1, c_2 \in C$ και $c_1 \prec c_2$ τότε η κλάση c_1 είναι υποκλάση της κλάσης c_2.*

5. $(P, \prec)$ *είναι μια σχέση μερικής διάταξης που ορίζεται για τις ακμές του* P *και αναπαριστά την ιεραρχία ιδιοτήτων ενός σχήματος RDF/S .Αν* $p_1, p_2 \in P$ *και* $p_1 \prec p_2$ *τότε η ιδιότητα* p_1 *είναι υπο-ιδιότητα της κλάσης* p_2.

Στο σχήμα 3.1 φαίνεται το απλούστερο δυνατό παράδειγμα που χρησιμοποιεί όλες τις δυνατότητες του RDF/S.



Σχήμα 3.1: Ένα απλό παράδειγμα RDF/S

Στο εξής θα σχεδιάζουμε τους γράφους RDF/S κάνοντας τις παρακάτω συμβάσεις, όπως και στο σχήμα:

1. Οι κλάσεις σχεδιάζονται ως σχήματα οβάλ τα οποία περικλείουν το όνομα της κλάσης.
2. Τα literals σχεδιάζονται ως παραλληλόγραμμα σχήματα τα οποία περικλείουν τον τύπο του literal.
3. Οι ιδιότητες σχεδιάζονται ως ονοματισμένα βέλη με λεπτή γραμμή τα οποία ξεκινούν από την κλάση που είναι πεδίο ορισμού της ιδιότητας και καταλήγουν στην κλάση ή στο literal που είναι πεδίο τιμών.
4. Οι σχέσεις της ιεραρχίας των κλάσεων σχεδιάζονται ως βέλη με παχιά γραμμή που ξεκινούν από την υποκλάση και καταλήγουν στην υπερκλάση.
5. Οι σχέσεις της ιεραρχίας των ιδιοτήτων σχεδιάζονται ως βέλη με παχιά γραμμή που ξεκινούν από την υπο-ιδιότητα και καταλήγουν στην υπερ-ιδιότητα.

3.1.2 Ορισμός υποσυνόλου και ισότητας για σχήματα RDF/S

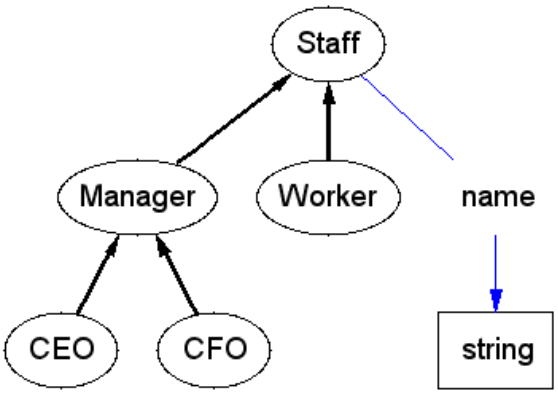
Για τους σκοπούς της περαιτέρω θεωρητικής ανάλυσης χρειάζεται να ορίσουμε τυπικά την έννοια του υποσυνόλου ενός σχήματος RDF/S βασιζόμενοι στον παραπάνω ορισμό του RDF/S ως γράφου:

Ορισμός 3.2 *Έστω δύο σχήματα RDF/S* $R_i = (C_i, L_i, P_i)$ *και* $R_j = (C_j, L_j, P_j)$. *Θα λέμε ότι το σχήμα* R_i *είναι υποσύνολο του σχήματος* R_j *και θα συμβολίζουμε με* $R_i \subseteq R_j$ *εάν ισχύουν όλα τα παρακάτω:*

1. $C_i \subseteq C_j$, δηλαδή το σύνολο των κλάσεων του πρώτου σχήματος είναι υποσύνολο των κλάσεων του δεύτερου.
2. $L_i \subseteq L_j$, δηλαδή το σύνολο των τύπων των λεκτικών του πρώτου σχήματος είναι υποσύνολο των κλάσεων του δεύτερου.
3. $P_i \subseteq P_j$, δηλαδή το σύνολο των ιδιοτήτων του πρώτου σχήματος είναι υποσύνολο των ιδιοτήτων του δεύτερου.
4. $\forall c_1, c_2 \in C_i \cap C_j$, εάν $c_1 \prec c_2$ στο σχήμα R_j, θα πρέπει το ίδιο να ισχύει και στο σχήμα R_i.
5. $\forall p_1, p_2 \in P_i \cap P_j$, εάν $p_1 \prec p_2$ στο σχήμα R_j, θα πρέπει το ίδιο να ισχύει και στο σχήμα R_i.

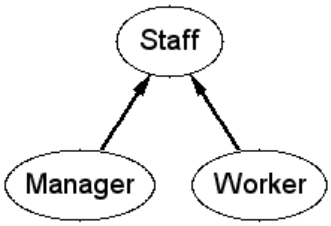
Με βάση τον παραπάνω ορισμό, ένα υποσύνολο ενός RDF/S σχήματος είναι ουσιαστικά ένας υπο-γράφος του RDF/S σχήματος στον οποίο συνεχίζουν να ισχύουν οι ίδιες σχέσεις διάταξης μεταξύ των κόμβων του.

Στο Σχήμα 3.2 φαίνεται ένα σχήμα RDF/S σε μορφή γράφου και στο Σχήμα 3.3 ένα υποσύνολό του. Στο υποσύνολο αυτό έχουμε επιλέξει μόνο τις κλάσεις `Staff`, `Manager` και `Worker`, οι οποίες αποτελούν υποσύνολο των κλάσεων του αρχικού σχήματος, άρα ικανοποιείται το πρώτο από τα παραπάνω κριτήρια. Επιπλέον, για τις κλάσεις αυτές ισχύουν οι ίδιες σχέσεις διάταξης όπως και στο αρχικό σχήμα, άρα ικανοποιείται και το τέταρτο κριτήριο. Τα υπόλοιπα κριτήρια είναι στο παράδειγμά μας τετριμμένα.

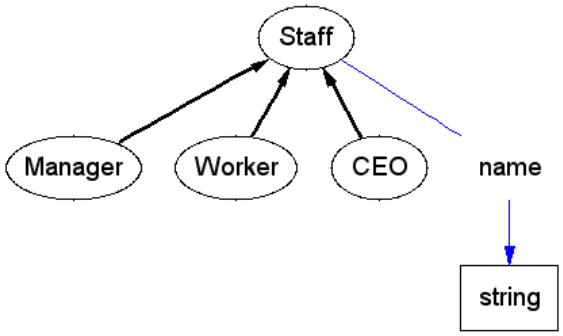


Σχήμα 3.2: Αρχικό σχήμα RDF/S

Αντίθετα, το σχήμα που φαίνεται στο Σχήμα 3.4 δεν είναι υποσύνολο του αρχικού σχήματος σύμφωνα με τον πιο πάνω ορισμό. Έχουμε επιλέξει τις κλάσεις `Staff`, `Manager`, `Worker` και `CEO`, οι οποίες αποτελούν υποσύνολο των κλάσεων του αρχικού σχήματος, άρα ικανοποιείται το πρώτο από τα κριτήρια του ορισμού. Οι ιδιότητες και οι τύποι των λεκτικών στο σχήμα αυτό είναι επίσης υποσύνολα του αρχικού σχήματος, όπως ορίζουν το τρίτο και το δεύτερο κριτήριο αντίστοιχα. Όμως στο αρχικό σχήμα αυτό έχουμε `CEO` $\prec$ `Manager` αλλά αυτό δεν ισχύει στο τροποποιημένο σχήμα, οπότε δεν ικανοποιείται το τέταρτο κριτήριο.



Σχήμα 3.3: Σχήμα το οποίο είναι υποσύνολο του αρχικού σχήματος



Σχήμα 3.4: Σχήμα το οποίο δεν είναι υποσύνολο του αρχικού σχήματος

3.2 Συμβατότητα RDF/S σχημάτων

3.2.1 Έννοια της 'συμβατότητας'

Ο τρόπος με τον οποίο θα ορίσουμε τη συμβατότητα δυο RDF/S σχημάτων δεν προέρχεται από κάποια μαθηματική αρχή, αλλά εξαρτάται από την ερμηνεία που δίνουμε στης δηλώσεις ενός σχήματος και από το νόημα των πράξεων που σκοπεύουμε να εφαρμόσουμε πάνω σε συμβατά σχήματα.

Στην εφαρμογή μας, τελικός σκοπός είναι να επιτρέψουμε μια σχετική ελευθερία στο σχήμα που θα χρησιμοποιεί κάθε κόμβος του δικτύου μας, διατηρώντας όμως τη δυνατότητα να κάνουμε ερωτήματα πάνω στο σύνολο των κόμβων και να παίρνουμε μια 'ενοποιημένη' απάντηση με τη χρήση ενός κατάλληλη αλγόριθμου απάντησης ερωτημάτων. Εφ'όσον θέλουμε να θέτουμε την ίδια ερώτηση σε όλους τους κόμβους του δικτύου, η ερώτηση αυτή θα πρέπει να έχει την ίδια 'σημασία' παντού. Θα ήταν, λ.χ. άτοπο να θεωρήσουμε συμβατά δύο σχήματα στα οποία η ίδια ιδιότητα έχει διαφορετικό πεδίο τιμών, διότι δε θα μπορούσαμε έτσι να πάρουμε ομοιόμορφα αποτελέσματα.

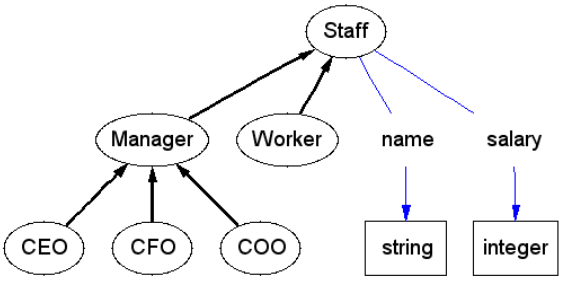
Ο ορισμός που δίνεται στη συνέχεια εξασφαλίζουν ότι με τη χρήση ενός κατάλληλου αλγορίθμου, τα αποτελέσματα θα μπορούν να είναι ενιαία.

3.2.2 Ορισμός της συμβατότητας

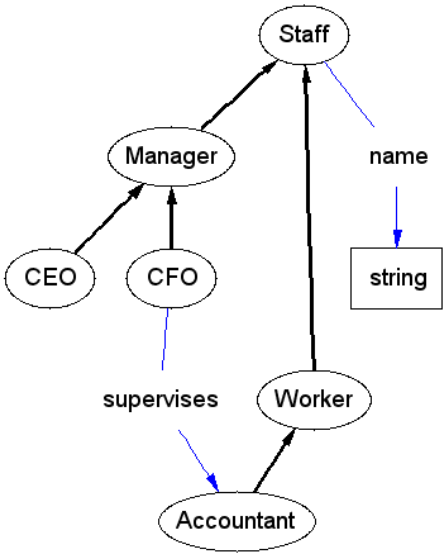
Ορισμός 3.3 *Έστω δύο σχήματα RDF/S* $R_i = (C_i, L_i, P_i)$ *και* $R_j = (C_j, L_j, P_j)$. *Θα λέμε ότι τα σχήμα* R_i, R_j *είναι συμβατά και θα συμβολίζουμε με* $R_i \sim R_j$ *αν και μόνο αν υπάρχει η τριάδα* $R_c = (C_i \cap C_j, L_i \cap L_j, P_i \cap P_j)$ *αναπαριστά ένα έγκυρο σχήμα RDF για το οποίο ισχύει επίσης* $R_c \subseteq R_i$ *και* $R_c \subseteq R_j$.

3.2.3 Παραδείγματα

Στα Σχήματα 3.5 και 3.6 απεικονίζονται δυο σχήματα RDF/S τα οποία παρ'όλο που με την πρώτη ματιά φαίνονται αρκετά διαφορετικά είναι τελικά συμβατά μεταξύ τους.



Σχήμα 3.5: Πρώτο συμβατό σχήμα (R_i)



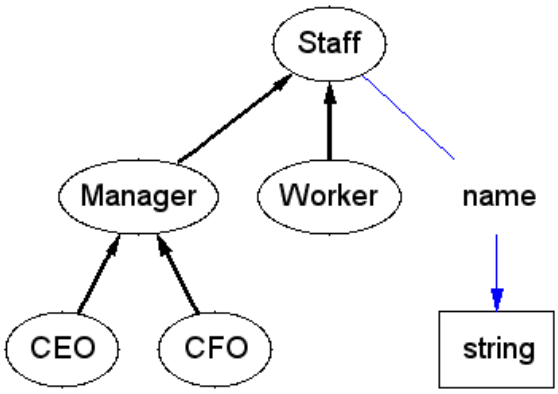
Σχήμα 3.6: Δεύτερο συμβατό σχήμα (R_j)

Πράγματι, σύμφωνα με τον προηγούμενο ορισμό έχουμε:

- Στο σύνολο $C_c = C_i \cap C_j$ ανήκουν οι κλάσεις `Staff`, `Worker`, `Manager`, `CEO`, `CFO`.
- Στο σύνολο $L_c = L_i \cap L_j$ ανήκει ο τύπος `string`

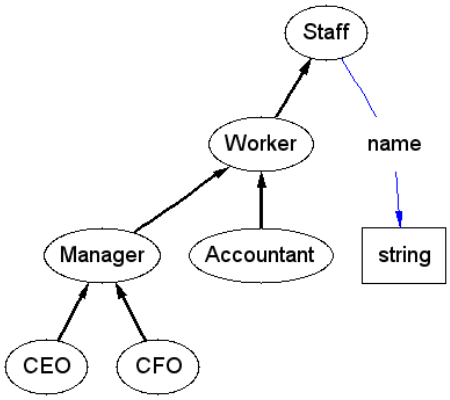
- Στο σύνολο $P_c = P_i \cap P_j$ ανήκει η ιδιότητα `name`

Το σχήμα R_c που ορίζεται από τις παρακάτω κλάσεις και ιδιότητες απεικονίζεται στο Σχήμα 3.7 και αποτελεί ένα υποσύνολο και των δύο συμβατών σχημάτων.



Σχήμα 3.7: Κοινό υποσύνολο R_c των R_i, R_j

Ας εξετάσουμε τώρα τη συμβατότητα του σχήματος R_j με το σχήμα R_k που απεικονίζεται στο Σχήμα 3.8.



Σχήμα 3.8: Τρίτο ασύμβατο σχήμα (R_k)

Στο παράδειγμα αυτό, αν και υπάρχουν ορισμένες κοινές κλάσεις, δε μπορούμε να βρούμε ένα κοινό υποσύνολο που να αποτελεί έγκυρο σχήμα RDF/S. Αυτό συμβαίνει επειδή στο R_k ισχύει `Manager` $\prec$ `Staff` ενώ στο R_j ισχύει `Manager` $\prec$ `Worker` $\prec$ `Staff`. Εάν σε ένα υποθετικό σχήμα R'_c θεωρούσαμε ότι ισχύει `Manager` $\prec$ `Worker`, τότε δεν θα ίσχυε $R'_c \subseteq R_j$. Εάν πάλι θεωρούσαμε ότι δεν ισχύει `Manager` $\prec$ `Worker`, τότε δεν θα ίσχυε $R'_c \subseteq R_k$.

Κεφάλαιο 4

Ανάλυση και σχεδίαση

Στο κεφάλαιο αυτό παρουσιάζεται η μελέτη που έγινε για να υλοποιηθεί το σύστημα. Αρχικά περιγράφεται η αρχιτεκτονική του συστήματος, απαριθμούνται τα διάφορα υποσυστήματα τα οποία το απαρτίζουν και περιγράφεται η λειτουργικότητα του καθενός καθώς και η μεταξύ τους επικοινωνία. Στη συνέχεια παρουσιάζεται η συνολική λειτουργικότητα της εφαρμογής και υπογραμμίζεται ο τρόπος συνεργασίας των υποσυστημάτων.

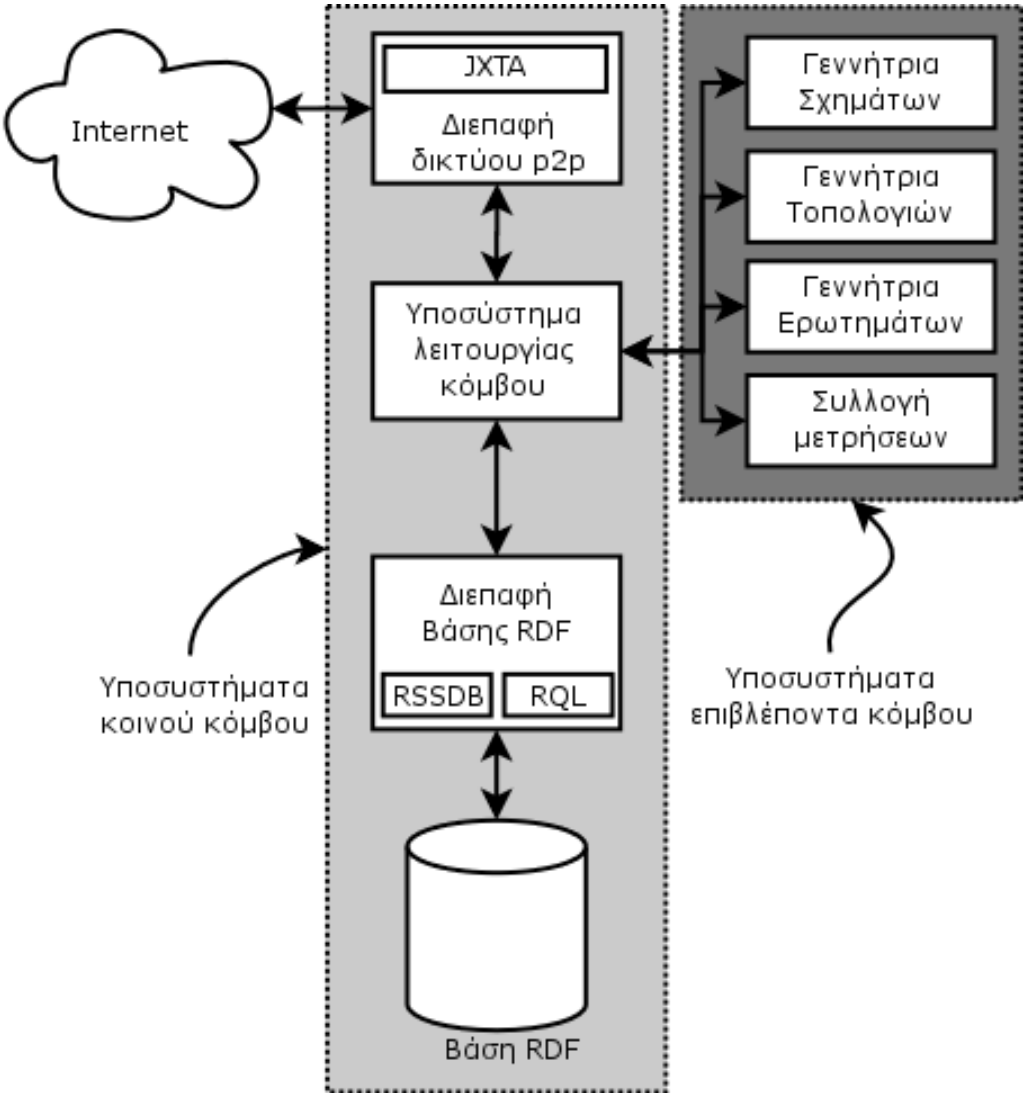
4.1 Περιγραφή αρχιτεκτονικής

Το σύστημα αποτελείται από τα εξής κύρια υποσυστήματα:

- Υποσύστημα διεπαφής με το δίκτυο ομότιμων κόμβων (p2p interface)
- Υποσύστημα διεπαφής με τη βάση RDF
- Υποσύστημα λειτουργίας του κόμβου (peer logic)
- Υποσύστημα δημιουργίας τυχαίων σχημάτων
- Υποσύστημα δημιουργίας τυχαίων τοπολογιών
- Υποσύστημα δημιουργίας τυχαίων ερωτημάτων
- Υποσύστημα συλλογής μετρήσεων
- Υποσύστημα διεπαφής χρήστη (user interface)
- Υποσύστημα εγκατάστασης του δικτύου (deployment)

Στο Σχήμα 4.1 φαίνονται τα παραπάνω υποσυστήματα και η μεταξύ τους επικοινωνία. Το σχήμα αφορά έναν μόνο από τους κόμβους του δικτύου. Στο συνολικό δίκτυο συνυπάρχουν πολλοί τέτοιοι κόμβοι οι οποίοι αποτελούνται από ανεξάρτητα στιγμιότυπα κάθε υποσυστήματος, εκτός από τη βάση δεδομένων την οποία μπορούν να μοιράζονται πολλοί κόμβοι, ο καθένας όμως δημιουργώντας το δικό του σχήμα.

Σημειώνουμε επίσης ότι στο σχήμα της αρχιτεκτονικής παρουσιάζονται δύο ομάδες υποσυστημάτων που περικλείονται από διακεκομμένες γραμμές. Η πρώτη ομάδα αφορά τα υποσυστήματα που αποτελούν υποχρεωτικά μέρος κάθε κόμβου του δικτύου, δηλαδή τα βασικά υποσυστήματα που χρειάζονται για τη σύνδεση με το δίκτυο p2p και τη βάση δεδομένων καθώς και για την απάντηση ερωτημάτων. Η δεύτερη ομάδα περιλαμβάνει υποσυστήματα τα οποία είναι χρειάζεται να είναι ενεργά μόνο στον κόμβο-συντονιστή του δικτύου, καθώς χρησιμεύουν στην αυτόματη παραγωγή δοκιμαστικών δεδομένων για τα σενάρια εξομοίωσης του συστήματος.



Σχήμα 4.1: Η αρχιτεκτονική του συστήματος

Στη συνέχεια περιγράφονται ένα προς ένα τα υποσυστήματα που αναφέρθηκαν παραπάνω και οι λειτουργίες που επιτελεί το καθένα.

4.1.1 Υποσύστημα διεπαφής με το δίκτυο ομότιμων κόμβων

Το υποσύστημα αυτό αναλαμβάνει το διαφανή χειρισμό των λειτουργιών που σχετίζονται με το δίκτυο ομότιμων κόμβων. Οι κυριότερες από τις λειτουργίες που υποστηρίζονται είναι οι εξής:

- Δημιουργία (εφ'όσον δεν υπάρχει ήδη) μιας ομάδας κόμβων (peer group) μέσω της οποίας ομαδοποιούνται όλοι οι κόμβοι που συμμετέχουν στο σύστημά μας. Με τον τρόπο αυτό εξασφαλίζεται η διαφοροποίηση από άλλους κόμβους που είναι τυχόν συνδεδεμένοι στο δίκτυο p2p για διαφορετικές εφαρμογές. Υπάρχει επίσης πρόβλεψη να τίθεται περιορισμός μέσω κωδικών ή άλλων συστημάτων ασφαλείας στην είσοδο των κόμβων στη συγκεκριμένη ομάδα, έτσι ώστε να αποκλείονται ανεπιθύμητοι κόμβοι.
- Σύνδεση με το δίκτυο p2p, δήλωση του παρόντος κόμβου και αίτηση συμμετοχής στην παραπάνω ομάδα κόμβων.
- Απαρίθμηση των κόμβων που συμμετέχουν στην παραπάνω ομάδα κόμβων. Η λειτουργία αυτή χρησιμοποιείται κυρίως από τον επιβλέποντα κόμβο για την αρχικοποίηση του δικτύου και τη διανομή σχημάτων, ερωτημάτων κτλ.
- Αποστολή και λήψη μηνυμάτων από/προς άλλους κόμβους του δικτύου. Μέσω των μηνυμάτων αυτών μπορούν να μεταδοθούν κάθε είδους δεδομένα, είτε αυτά είναι σε μορφή κειμένου (XML) είτε δυαδικά αρχεία δεδομένων (binary files).

Για καθεμιά από τις παραπάνω λειτουργίες παρέχεται από το υποσύστημα η αντίστοιχη διεπαφή. Ας σημειωθεί στο σημείο αυτό πως μόνο αυτό το υποσύστημα εξαρτάται από την υλοποίηση του δικτύου p2p. Κατά συνέπεια, με αλλαγή μόνο της υλοποίησης του συγκεκριμένου υποσυστήματος, η εφαρμογή μπορεί εύκολα να προσαρμοστεί ούτως ώστε να είναι δυνατόν να συνδεθεί σε κάποιο δίκτυο p2p διαφορετικό από αυτό που υλοποιούμε στην παρούσα εργασία. Στην περίπτωση αυτή, χρειάζεται προφανώς το καινούριο δίκτυο p2p να είναι σε θέση να υποστηρίξει τις παραπάνω λειτουργίες.

4.1.2 Υποσύστημα διεπαφής με τη βάση RDF

Το υποσύστημα αυτό αναλαμβάνει το διαφανή χειρισμό των λειτουργιών που σχετίζονται με τη βάση RDF. Χρησιμοποιείται για τη μόνιμη αποθήκευση RDF και την εκτέλεση ερωτημάτων από τους κόμβους του δικτύου ενώ ταυτόχρονα υποβοηθά το υποσύστημα δημιουργίας τυχαίων σχημάτων.

Οι κυριότερες από τις λειτουργίες που υποστηρίζονται είναι οι εξής:

- Αρχικοποίηση της σύνδεσης με τον εξυπηρετητή ΒΔ.
- Δημιουργία (εφ'όσον δεν υπάρχει ήδη) μιας βάσης για την αποθήκευση των σχημάτων RDF/S και των δεδομένων RDF ενός κόμβου.
- Αποθήκευση σχημάτων RDF/S και δεδομένων RDF στη βάση του κόμβου.

- Εκτέλεση ερωτημάτων πάνω στα σχήμα και τα δεδομένα με χρήση της γλώσσας RQL.

- Μετατροπή των αποτελεσμάτων των ερωτημάτων RQL σε κατάλληλες δομές δεδομένων που μπορούν να χρησιμοποιηθούν εύκολα από τα υπόλοιπα υποσυστήματα.

Για καθεμιά από τις παραπάνω λειτουργίες παρέχεται από το υποσύστημα η αντίστοιχη διεπαφή. Όπως και στο υποσύστημα διεπαφής p2p, όλες οι λεπτομέρειες της αποθήκευσης και επεξεργασίας RDF σχημάτων και δεδομένων ενθυλακώνονται στο παρόν υποσύστημα. Έτσι, είναι κι εδώ εφικτό να αλλάξει ο τρόπος αποθήκευσης και πιθανόν η γλώσσα περιγραφής των ερωτημάτων τροποποιώντας μόνο το αντίστοιχο υποσύστημα, εφ'όσον βέβαια η νέα υλοποίηση υποστηρίζει επαρκώς όλες τις απαιτούμενες λειτουργίες.

4.1.3 Υποσύστημα λειτουργίας του κόμβου

Πρόκειται για το υποσύστημα που συντονίζει όλα τα υπόλοιπα υποσυστήματα κάθε κόμβου, και μέσω αυτού υλοποιείται ουσιαστικά η λειτουργικότητα που έχουν οι κόμβοι του δικτύου, δηλαδή η λογική της εφαρμογής μας (application logic).

Στην τρέχουσα υλοποίηση, το υποσύστημα εκτελεί τις ελάχιστες λειτουργίες που είναι απαραίτητες για να λειτουργήσει σωστά το δίκτυο και να δίνονται απαντήσεις σε ερωτήματα, καθώς σκοπός μας είναι η δημιουργία ενός απλού δικτύου για δοκιμαστικούς σκοπούς. Σε περίπτωση που το σύστημά μας επεκταθεί για να υποστηρίξει πιο πολύπλοκες εφαρμογές, το υποσύστημα αυτό πρέπει να τροποποιηθεί κατάλληλα. Για παράδειγμα, μπορούν να ενσωματωθούν προηγμένοι αλγόριθμοι επεξεργασίας, απάντησης και δρομολόγησης ερωτημάτων (query processing, answering and routing).

Ο ρόλος του υποσυστήματος αυτού είναι διαφορετικός ανάλογα με το ρόλο του κόμβου (κοινός ή επιβλέπων κόμβος). Ένα μέρος της λειτουργικότητας είναι κοινό για όλους τους κόμβους και περιλαμβάνει την αρχικοποίηση των υποσυστημάτων (κυρίως των διεπαφών RDF και p2p) και απάντηση και την προώθηση ερωτημάτων. Οι υπόλοιπες λειτουργίες αφορούν τη διαχείριση του δικτύου και εκτελούνται μόνο από τον επιβλέπων κόμβο.

Η λειτουργικότητα του παρόντος υποσυστήματος παρουσιάζεται αναλυτικότερα στην Ενότητα 4.2.

4.1.4 Υποσύστημα δημιουργίας τυχαίων σχημάτων

Το υποσύστημα αυτό παίρνει ως είσοδο του ένα σχήμα RDF/S και τροποποιεί το σχήμα ώστε να παράγει πολλά τυχαία σχήματα. Τα σχήματα αυτά είναι συμβατά με το αρχικό σχήμα καθώς και μεταξύ τους σύμφωνα με τον ορισμό που δόθηκε στην ενότητα 3.2.

Για την εκτέλεση της παραπάνω λειτουργίας συνεργάζεται επίσης το υποσύστημα διεπαφής με τη βάση RDF, μέσω του οποίου μπορούμε να επεξεργαστούμε αποτελεσματικά ένα σχήμα RDF/S.

4.1.5 Υποσύστημα δημιουργίας τυχαίων τοπολογιών

Το υποσύστημα αυτό παίρνει ως είσοδο του μια λίστα με τους κόμβους που απαρτίζουν το δίκτυο p2p και δημιουργεί μια τυχαία τοπολογία, μέσω της οποίας μεταδίδονται τα ερωτήματα στο δίκτυο.

Η τοπολογία που δημιουργείται είναι ένας κατευθυνόμενος άκυκλος γράφος (Directed Acyclic Graph - DAG). Στη θέση της ρίζας αυτού του γράφου (δηλαδή του κόμβου προς τον οποίον δεν κατευθύνεται καμιά ακμή) τοποθετείται πάντα ο επιβλέπων κόμβος, αφού από τον κόμβο αυτό ξεκινά η μετάδοση των ερωτημάτων.

4.1.6 Υποσύστημα δημιουργίας τυχαίων ερωτημάτων

Το υποσύστημα αυτό δημιουργεί τυχαία ερωτήματα RQL τα οποία χρησιμοποιούνται για τον έλεγχο και τη μέτρηση της απόδοσης του συστήματός μας. Τα δημιουργούμενα ερωτήματα πρέπει κατ'αρχάς να είναι ορθά συντακτικά σύμφωνα με τις αρχές της RQL και παράγονται με συνδυασμό διαφόρων στοιχείων της γλώσσας ώστε να προκύπτουν ερωτήματα διαφορετικής πολυπλοκότητας. Κατά την παραγωγή των ερωτημάτων λαμβάνονται επίσης υπ'όψη στοιχεία από το αρχικό σχήμα ώστε τα ερωτήματα να σχετίζονται με το σχήμα που διαθέτει κάθε κόμβος.

4.1.7 Υποσύστημα συλλογής μετρήσεων

Το υποσύστημα αυτό παρακολουθεί τη λειτουργία του δικτύου κατά τη φάση της συνεχούς αυτοματοποιημένης υποβολής ερωτημάτων και συλλέγει στατιστικά. Ορισμένες από τις μετρήσεις που γίνονται είναι οι παρακάτω:

- Ο χρόνος εκτέλεσης του ερωτήματος σε κάθε κόμβο.
- Ο χρόνος μετάβασης του ερωτήματος από κόμβο σε κόμβο.
- Ο χρόνος μετάβασης της απάντησης ενός ερωτήματος από τον κόμβο που το εκτέλεσε έως τον επιβλέποντα κόμβο.
- Ο αριθμός των βημάτων (hops) από κόμβο σε κόμβο που διένυσε ένα μήνυμα μέχρι να βρεθεί η πρώτη μη κενή απάντηση.

4.1.8 Υποσύστημα διεπαφής χρήστη

Μέσω του υποσυστήματος διεπαφής χρήστη παρουσιάζονται στον χρήστη:

- Διαχειριστικά στοιχεία για τη λειτουργία του δικτύου.
- Πληροφορίες που χρησιμεύουν στην εκσφαλμάτωση (debugging).
- Τα στατιστικά στοιχεία που παρέχει το υποσύστημα συλλογής μετρήσεων.

Δίνεται επίσης η δυνατότητα στο χρήστη να χειριστεί το σύστημα, δηλαδή :

- να επιλέξει το αρχικό σχήμα που θα χρησιμοποιηθεί από το υποσύστημα δημιουργίας τυχαίων σχημάτων
- να εκκινήσει και να σταματήσει τη διαδικασία μετρήσεων, δηλαδή τον κύκλο αυτόματης αποστολής σχημάτων και ερωτημάτων στους κόμβους.
- να αποθηκεύσει τα στατιστικά στοιχεία για περαιτέρω επεξεργασία.
- να ρυθμίσει ορισμένες παραμέτρους των υποσυστημάτων δημιουργίας τυχαίων σχημάτων, τοπολογιών και ερωτημάτων.

Οι πληροφορίες παρουσιάζονται με μεταβλητό επίπεδο λεπτομέρειας (verbosity) ρυθμιζόμενο από το χρήστη. Για παράδειγμα, ο χρήστης μπορεί να επιλέξει να βλέπει μόνο τις βασικές πληροφορίες για την πρόοδό της λειτουργίας του συστήματος ή να παρακολουθήσει αναλυτική περιγραφή κάθε βήματος που γίνεται, σε περίπτωση που χρειαστεί να γίνει εκσφαλμάτωση.

Ειδικά για τη διεπαφή χρήστη των κανονικών κόμβων του δικτύου, οι οποίοι πιθανόν να εγκατασταθούν σε μηχανήματα που δε διαθέτουν οθόνη και πληκτρολόγιο, έχει προβλεφθεί η απενεργοποίηση του γραφικού περιβάλλοντος και η προβολή των πληροφοριών με μορφή κειμένου στη γραμμή εντολών, έτσι ώστε να είναι δυνατή η εξ'αποστάσεως θέση σε λειτουργία και επιτήρηση του συστήματος.

4.1.9 Υποσύστημα εγκατάστασης του δικτύου

Το υποσύστημα αυτό δεν περιλαμβάνεται σε αυτά που είναι ενεργά κατά το χρόνο λειτουργίας του συστήματος, καθώς είναι υπεύθυνο για τη θέση του συστήματος σε λειτουργία (deployment). Με τη βοήθεια του υποσυστήματος αυτού μπορεί να εγκατασταθεί σε ένα υπολογιστικό σύστημα ένα σύνολο κόμβων οι οποίοι να τεθούν αυτόματα σε λειτουργία. Με τον τρόπο αυτό μπορεί εύκολα και με ελάχιστη ανθρώπινη παρέμβαση να προγραμματιστεί ένα σύνολο πειραμάτων με το δίκτυο και να συλλεχθούν αυτόματα τα αντίστοιχα στατιστικά στοιχεία.

4.2 Περιγραφή Λειτουργικότητας

Στην ενότητα αυτή περιγράφονται οι λειτουργίες του συστήματός και ο τρόπος με τον οποίον αυτές συντονίζονται από το υποσύστημα λειτουργίας του κόμβου.

4.2.1 Λειτουργικότητα του κοινού κόμβου

Με την έναρξη της λειτουργίας του κόμβου αρχικοποιείται το υποσύστημα διεπαφής p2p και γίνεται αίτηση σύνδεσης με το δίκτυο p2p. Αρχικοποιείται επίσης το υποσύστημα διεπαφής RDF και γίνεται αίτησης σύνδεσης με τη βάση RDF. Τα στοιχεία αρχικοποίησης, δηλαδή το όνομα του κόμβου και του δικτύου p2p καθώς και η θέση και το όνομα της βάσης RDF παρέχονται από το υποσύστημα εγκατάστασης του δικτύου μέσω ενός αρχείου ρυθμίσεων

(configuration file). Σε περίπτωση που κάποια από τις παραπάνω λειτουργίες αποτύχουν, ο κόμβος τερματίζει τη λειτουργία του και παρουσιάζονται όλες οι σχετικές πληροφορίες που μπορούν να διευκολύνουν την εκσφαλμάτωση.

Στη συνέχεια ο κόμβος αναμένει για εισερχόμενα συνδέσεις από άλλους κόμβους. Για κάθε μια από τις συνδέσεις αυτές εκκινείται ξεχωριστή διεργασία (thread) εξυπηρέτησης μηνυμάτων για το χειρισμό των μηνυμάτων που θα ληφθούν από τη σύνδεση αυτή και ο κόμβος συνεχίζει να αναμένει για νέες συνδέσεις.

Η διεργασία εξυπηρέτησης μηνυμάτων αναμένει για εισερχόμενα μηνύματα τα οποία και εξυπηρετεί ανάλογα με τον τύπο τους. Εάν το μήνυμα αφορά αποθήκευση νέων σχημάτων ή δεδομένων RDF, εκτελούνται οι αντίστοιχες λειτουργίες μέσω του υποσυστήματος διεπαφής RDF. Εάν το μήνυμα αφορά ερώτημα RQL, το ερώτημα εκτελείται μέσω του υποσυστήματος διεπαφής RDF. Με τη βοήθεια του υποσυστήματος διεπαφής p2p τα αντίστοιχα αποτελέσματα επιστρέφονται στον ερωτώντα κόμβο και το ερώτημα προωθείται και στους γειτονικούς κόμβους. Τέλος, για τα λοιπά διαχειριστικά μηνύματα εκτελούνται οι αντίστοιχες διαχειριστικές λειτουργίες (π.χ. ορισμός γειτονικών κόμβων στην τοπολογία του δικτύου).

Η παραπάνω διαδικασία παρουσιάζεται με μορφή διαγραμμάτων ροής (flowchart) στα Σχήμα 4.2 και 4.3.

4.2.2 Λειτουργικότητα του επιβλέποντα κόμβου

Με την έναρξη της λειτουργίας του κόμβου εκτελείται η αρχικοποίηση των υποσυστημάτων διεπαφής όπως και στον κοινό κόμβο.

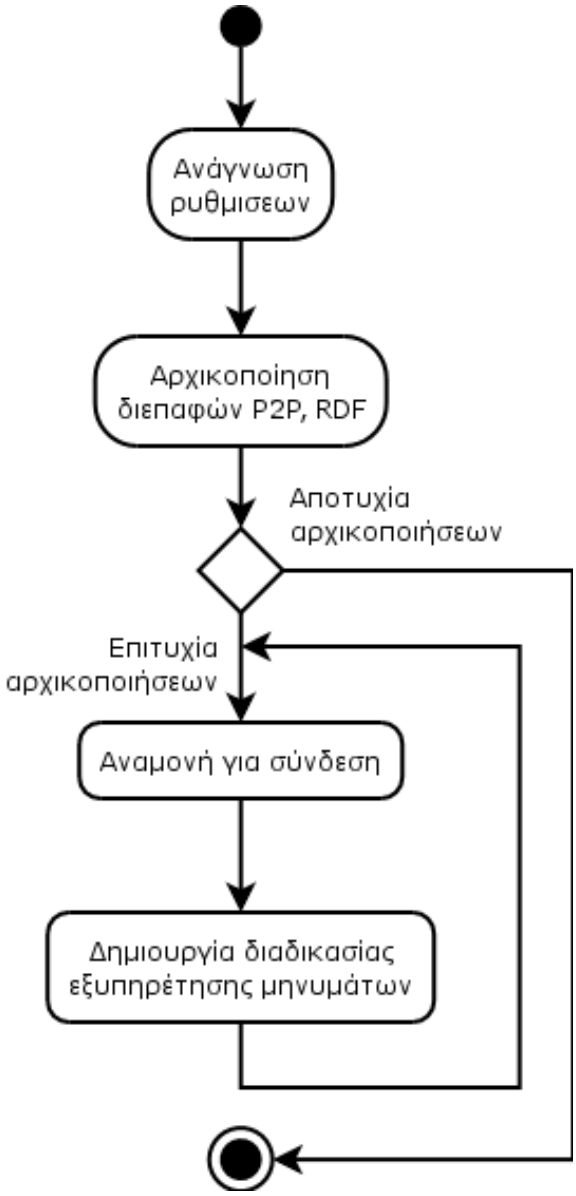
Στη συνέχεια, απαριθμούνται οι κόμβων που ανήκουν στο δίκτυο μέσω του υποσυστήματος διεπαφής p2p. Η λίστα των κόμβων αποστέλλεται στο υποσύστημα δημιουργίας τυχαίων τοπολογιών και παράγεται μια τοπολογία η οποία και αποστέλλεται σε κάθε κόμβο οι πληροφορίες για τους γειτονικούς του κόμβους.

Κατόπιν αποθηκεύεται στη βάση RDF το πρότυπο σχήμα του συστήματος, το οποίο έχει οριστεί από το χρήστη μέσω του υποσυστήματος εγκατάστασης του δικτύου. Το σχήμα αυτό επεξεργάζεται από το υποσύστημα δημιουργίας τυχαίων σχημάτων και παράγεται ένα σύνολο από τυχαία παράγωγα σχήματα, το καθένα εκ των οποίων αποστέλλεται σε έναν κόμβο του δικτύου.

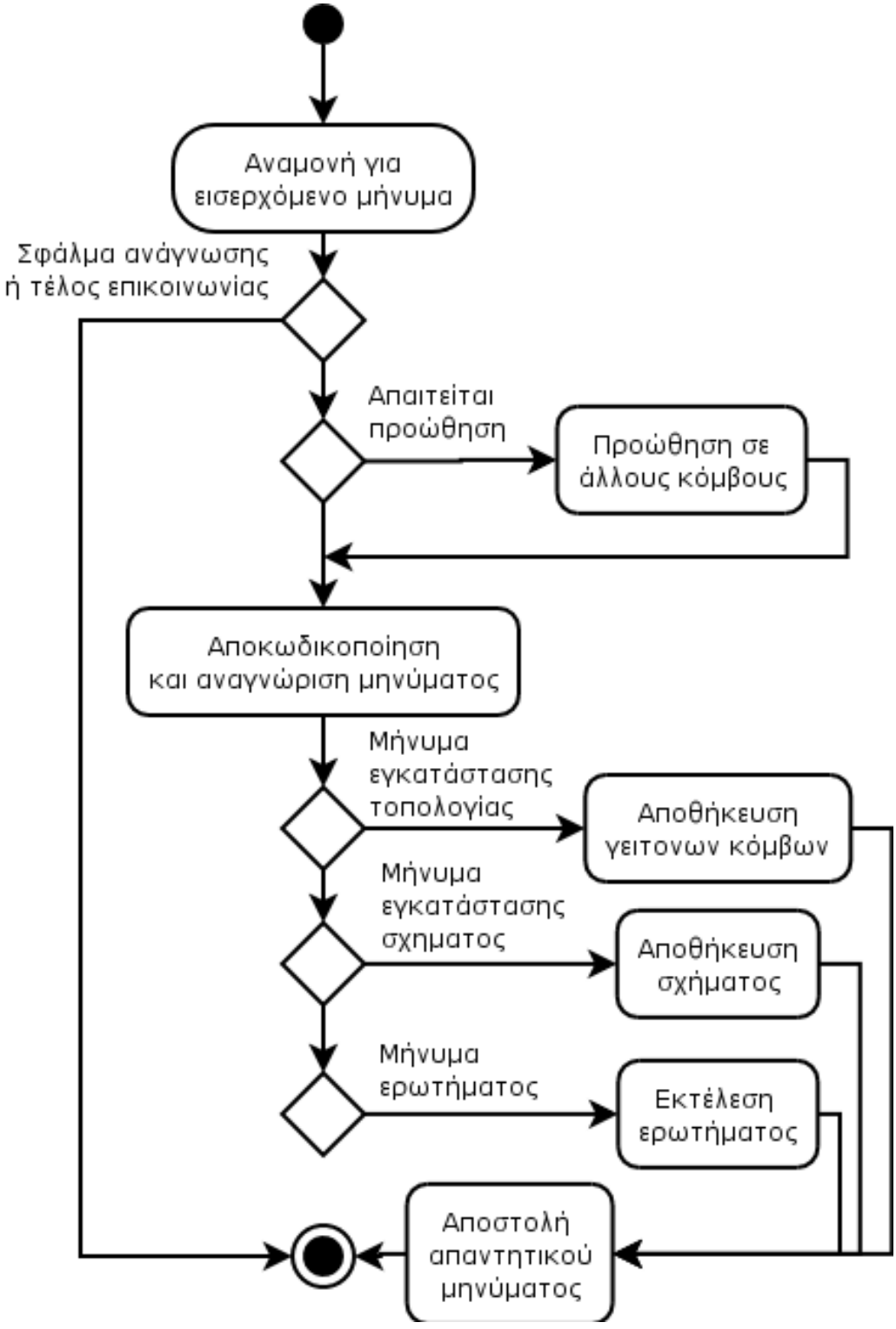
Τέλος, ο κόμβος εισέρχεται σε έναν βρόχο συνεχούς δημιουργία τυχαίων ερωτημάτων (μέσω του υποσυστήματος δημιουργίας τυχαίων ερωτημάτων) και αποστολής τους στο δίκτυο προς εκτέλεση. Παράλληλα, το υποσύστημα συλλογής μετρήσεων καταγράφει τα απαραίτητα στατιστικά στοιχεία. Η έξοδος από τον βρόχο μπορεί να γίνει είτε κατ'απαίτηση του χρήστη είτε μετά από ολοκλήρωση προκαθορισμένου αριθμού επαναλήψεων.

Καθ'όλη της διαδικασίας παρουσιάζονται στο χρήστη οι απαραίτητες πληροφορίες προόδου μέσω του υποσυστήματος διεπαφής χρήστη.

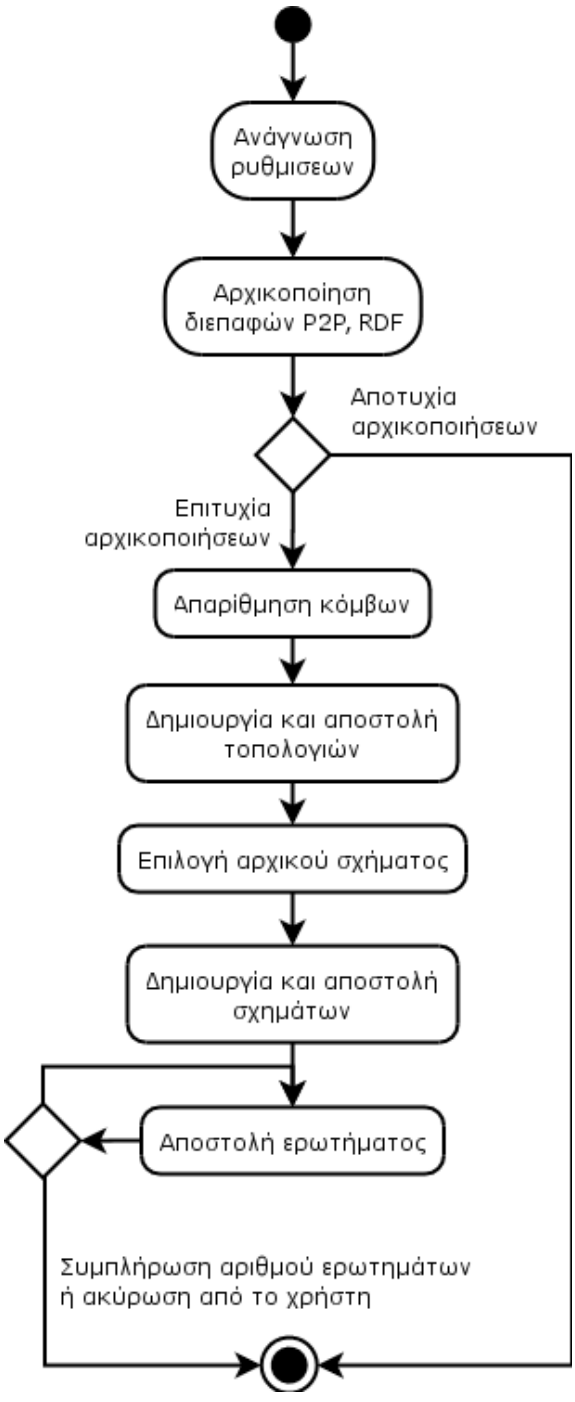
Η παραπάνω διαδικασία παρουσιάζεται με μορφή διαγράμματος ροής (flowchart) στο Σχήμα 4.4.



Σχήμα 4.2: Διάγραμμα ροής του κοινού κόμβου



Σχήμα 4.3: Διάγραμμα ροής της διαδικασίας εξυπηρέτησης μηνυμάτων



Σχήμα 4.4: Διάγραμμα ροής του επιβλέποντα κόμβου

Κεφάλαιο 5

Υλοποίηση

Στο κεφάλαιο αυτό περιγράφεται η υλοποίηση του συστήματος με βάση τη μελέτη που αναφέρθηκε στο προηγούμενο κεφάλαιο. Αρχικά παρουσιάζεται η πλατφόρμα υλοποίησης και λειτουργίας του συστήματος και τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν. Στη συνέχεια περιγράφονται για κάθε υποσύστημα ξεχωριστά οι λεπτομέρειες υλοποίησής του, οι αλγόριθμοι που αναπτύχθηκαν και οι δομή του κώδικα της εφαρμογής.

5.1 Πλατφόρμες και προγραμματιστικά εργαλεία

5.1.1 Γλώσσα προγραμματισμού

Το σύστημα υλοποιήθηκε χρησιμοποιώντας τη γλώσσα προγραμματισμού Java. Η Javaεπιλέχθηκε διότι είναι μια σύγχρονη γλώσσα αντικειμενοστραφούς προγραμματισμού και μας βοηθά να γράψουμε εύκολα κατανοητό και επαναχρησιμοποιήσιμο κώδικα. Επιπλέον, οι βιβλιοθήκες που χρησιμοποιήσαμε για την διαχείριση δεδομένων RDF και τη σύνδεση με το δίκτυο ομότιμων κόμβων ήταν κι αυτές υλοποιημένες σε Java. Τέλος, η ίδια η Java διαθέτει μια πλούσια βιβλιοθήκη κλάσεων (standard class library) η οποία περιέχει πολλές κλάσεις χρήσιμες για την εφαρμογή μας οι οποίες αφορούν δομές δεδομένων, παραλληλία (threads) και ανάγνωση δεδομένων XML (XML parsing).

5.1.2 Λειτουργικό σύστημα

Το σύστημα αναπτύχθηκε σε λειτουργικό σύστημα GNU/Linux. Το Linux επιλέχθηκε λόγω της σταθερότητάς του και των δυνατοτήτων αυτοματοποίησης που προσφέρει. Επιπλέον, ορισμένες από τις βιβλιοθήκες που χρησιμοποιήσαμε υποστήριζαν μόνο λειτουργικά τύπου Unix.

5.1.3 Προγραμματιστικά εργαλεία

Για τη μεταγλώττιση του πηγαίου κώδικα σε Javakαι την εκτέλεση του συστήματος χρησιμοποιήθηκε το JavaDevelopment Kit (JDK) της εταιρίας Sun Microsystems™. Για τη

μεταγλώττιση ορισμένων βιβλιοθηκών που ήταν γραμμένες σε γλώσσα C χρησιμοποιήθηκε ο μεταγλωττιστής `gcc` ο οποίος είναι διαθέσιμος μαζί με το λειτουργικό GNU/Linux.

Για την ανάπτυξη και την εκσφαλμάτωση του πηγαίου κώδικα σε Javaχρησιμοποιήθηκε το ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment - IDE) Eclipse.

Για την αυτόματη παραγωγή τεκμηρίωσης από τα σχόλια του πηγαίου κώδικα χρησιμοποιήθηκε το εργαλείο `javadoc` το οποίο αποτελεί μέρος του JavaDevelopment Kit.

5.1.4 Πλατφόρμα δικτύου ομότιμων κόμβων

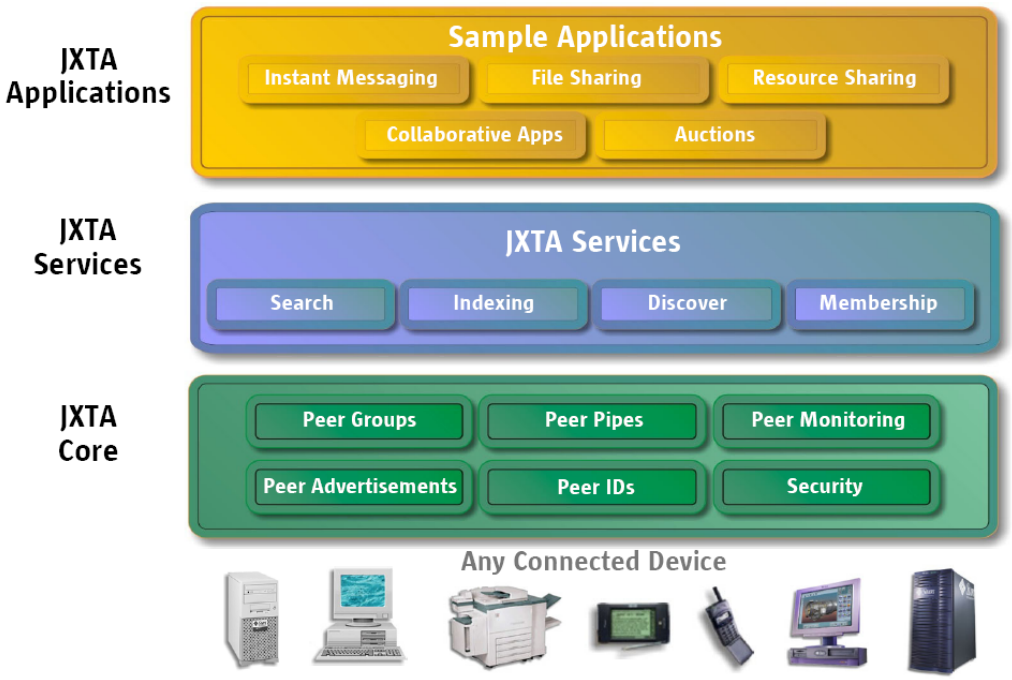
Το δίκτυο ομότιμων κόμβων της εφαρμογής μας υλοποιήθηκε με τη βοήθεια της βιβλιοθήκης JXTA [Gon01].

Το JXTA είναι κατ'αρχάς ένα σύνολο ανοιχτών και επεκτάσιμων πρωτοκόλλων για την επικοινωνία συσκευών σε δίκτυα ομότιμων κόμβων. Τα πρωτόκολλα αυτά ορίζουν μια κοινή γλώσσα μέσω της οποίας κάθε κόμβος που συνδέεται στο δίκτυο μπορεί να συνεννοηθεί και να ανταλλάξει δεδομένα με άλλους κόμβους του δικτύου. Τα πρωτόκολλα αυτά:

- Είναι ανεξάρτητα από το είδος των κόμβων που επικοινωνούν: μπορεί να πρόκειται για απλούς υπολογιστές ή για συσκευές όπως κινητά τηλέφωνα, PDAs κτλ.
- Είναι ανεξάρτητα από τη γλώσσα προγραμματισμού που χρησιμοποιήθηκε για την ανάπτυξη του λογισμικού κάθε κόμβου. Η πρότυπη υλοποίηση του JXTA έχει αναπτυχθεί σε γλώσσα Javaαλλά παράλληλα εξελίσσονται υλοποιήσεις σε άλλες γλώσσες όπως C, C++ και Perl.
- Είναι ανεξάρτητα από τη μέθοδο επικοινωνίας (message transport) μεταξύ των κόμβων. Τις περισσότερες φορές η επικοινωνία των κόμβων γίνεται μέσω του διαδικτύου και των πρωτοκόλλων TCP/IP αλλά μπορούν να υποστηριχθούν και άλλες μέθοδοι επικοινωνίας, π.χ. Bluetooth
- Βασίζονται σε ανοιχτές μορφές δεδομένων όπως π.χ. η γλώσσα XML

Σκοπός της ανάπτυξης των παραπάνω πρωτοκόλλων είναι η προτυποποίηση των βασικών λειτουργικών ενός δικτύου p2p, δηλαδή του του τρόπου με τον οποίο οι κόμβοι του δικτύου:

- αναζητούν και ανακαλύπτουν ο ένας τον άλλον.
- αυτο-οργανώνωνται σε ομάδες κόμβων (peer groups).
- ανακοινώνουν και ανακαλύπτουν τις υπηρεσίες που προσφέρονται μέσω του δικτύου.
- επικοινωνούν ο ένας με τον άλλον.
- παρακολουθούν ο ένας τη λειτουργία του άλλου.



Σχήμα 5.1: Η αρχιτεκτονική του JXTA

Στο Σχήμα 5.1 φαίνεται η αρχιτεκτονική του JXTA. Στο κατώτερο επίπεδο, υπάρχει ο πυρήνας του JXTA (JXTA core) όπου περιλαμβάνονται οι λειτουργίες που είναι απαραίτητες σε κάθε δίκτυο ομότιμων κόμβων, όπως η ονοματοδοσία (Peer IDs), η οργάνωση των κόμβων σε ομάδες (Peer Groups), ο μηχανισμός 'ανακοινώσεων' (Peer Advertisements) και ο μηχανισμός επικοινωνίας (Peer Pipes). Το αμέσως επόμενο επίπεδο είναι το επίπεδο υπηρεσιών (JXTA Services), στο οποίο περιλαμβάνονται λειτουργίες που είναι χρήσιμες (αλλά όχι απολύτως απαραίτητες) για τη λειτουργία των περισσότερων p2p δικτύων. Τέτοιες υπηρεσίες είναι συνήθως υπηρεσίες αναζήτησης (searching), καταλόγου (directory), κατανεμημένης αποθήκευσης (distributed storage) κ.α. Τέλος, στο ανώτερο επίπεδο ανήκουν οι εφαρμογές που βασίζονται σε δίκτυα p2p: ανταλλαγή αρχείων (file sharing), ανταλλαγή μηνυμάτων (instant messaging) κ.α. Στο επίπεδο αυτό ανήκει και η δική μας εφαρμογή.

Το όφελος από τη χρήση του JXTA, σε αντίθεση με την ανάπτυξη από το μηδέν μιας πλατφόρμας p2p, είναι προφανές. Σε πρώτη φάση, αποφεύγεται η επανα-υλοποίηση πολύπλοκων λειτουργιών που υλοποιούνται ήδη με αποτελεσματικό και δοκιμασμένο τρόπο από το JXTA. Μια από τις σημαντικότερες ευκολίες που μας προσφέρεται από το JXTA είναι η δυνατότητα επικοινωνίας μεταξύ δυο κόμβων που διαχωρίζονται από διακομιστές μεσολάβησης (proxies) ή τείχη προστασίας (firewalls). Επιπλέον με τη χρήση του JXTA το σύστημά μας καθίσταται επεκτάσιμο και είναι δυνατόν να προσπελαστεί από κάθε είδους κόμβο που ακολουθεί τα πρωτόκολλα του JXTA ή να γίνει μέρος ενός μεγαλύτερου δικτύου ομότιμων κόμβων.

5.1.5 Εργαλεία διαχείρισης RDF

Όπως αναφέρθηκε και στην Ενότητα 2.1.2, η αποθήκευση και η εκτέλεση ερωτήσεων σε RDF σχήματα και δεδομένα γίνεται με τη βοήθεια του συστήματος RDFSuite [ACK+01]. Το RDFSuite αναπτύχθηκε από το Ινστιτούτο Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνάς (ICS-FORTH) και αποτελείται από τα εξής επιμέρους εργαλεία:

- Validating RDF Parser (VRP), ένας parser (συντακτικός αναλυτής) ο οποίος μπορεί να εξετάσει τη σημασιολογική ορθότητα ενός σχήματος RDF καθώς και τη συμφωνία περιγραφών RDF με το σχήμα που υποτίθεται ότι ακολουθούν (validation). Ο VRP παρέχει επίσης τη δυνατότητα μετατροπής ενός RDF αρχείου σε μια δομή δεδομένων στη μνήμη, προσβάσιμη μέσω κατάλληλων διεπαφών (APIs).

- RDF Schema Specific DataBase (RSSDB), ένα σύστημα το οποίο αποθηκεύει δεδομένα RDF σε αντικειμενοστραφείς σχεσιακές βάσεις δεδομένων (Object-Relational Databases, πρότυπο SQL3). Το σχήμα το οποίο χρησιμοποιείται για την αποθήκευση των δεδομένων στη βάση προκύπτει από το σχήμα RDF/S το οποίο ακολουθούν τα δεδομένα RDF.

- RDF Query Language (RQL) Interpreter, ένας διερμηνέας για τη γλώσσα RQL που περιγράφηκε στην ενότητα 2.1.2. Η εκτέλεση των ερωτημάτων RQL γίνεται αφού τα δεδομένα αποθηκευθούν σε μια βάση δεδομένων με τη βοήθεια του προηγούμενου εργαλείου.

Η βάση δεδομένων που χρησιμοποιήσαμε για την υλοποίηση του συστήματός μας είναι η PostgreSQL, μια σύγχρονη αντικειμενοστραφής σχεσιακή βάση δεδομένων η οποία συνεργάζεται άριστα με το RDFSuite.

5.1.6 Πλατφόρμα εκτέλεσης

Το σύστημα έχει σχεδιαστεί να λειτουργήσει σε περιβάλλον GNU/Linux. Μιας και κανένα τμήμα του συστήματος δεν εξαρτάται από ιδιαιτερότητες μια συγκεκριμένης διανομής (distribution) του Linux, το σύστημα είναι φορητό (portable) σε όλες τις διανομές Linux. Εκτιμούμε επιπλέον ότι το σύστημα μπορεί να μεταφερθεί χωρίς καμιά τροποποίηση ή με ελάχιστες τροποποιήσεις σε άλλα λειτουργικά συστήματα τύπου Unix.

Για τη λειτουργία του συστήματος απαιτείται να είναι εγκατεστημένο το JavaRuntime Environment (JRE) το οποίο είναι ελεύθερα διαθέσιμο από την εταιρία Sun Microsystems™.

5.1.7 Εκδόσεις και διαθεσιμότητα των εργαλείων

Στον Πίνακα 5.1 παρουσιάζονται οι αριθμοί εκδόσεων (version numbers) των εργαλείων που αναφέρθηκαν στις προηγούμενες παραγράφους καθώς και οι δικτυακοί τόποι από τους οποίους διατίθενται προς μεταφόρτωση τα εργαλεία αυτά. Οι ανάπτυξη και οι δοκιμές του συστήματος έγιναν με τις εκδόσεις των εργαλείων που αναφέρονται παρακάτω. Για τη χρήση

Πίνακας 5.1: Εκδόσεις και διαθεσιμότητα εργαλείων

Εργαλείο	Έκδοση	Ιστότοπος
JavaDevelopment Kit (JDK)	`java version "1.4.2_06"`	http://java.sun.com/j2se
GNU C Compiler (GCC)	`gcc version 3.3.3`	http://gcc.gnu.org/
JXTA	`JXTA 2.3.1`	http://www.jxta.org/
ICS-FORTH RDFSuite	`RSSDB v.2.0, RQL v.2.1`	http://139.91.183.30:9090/RDF/
PostgreSQL	`postgres (PostgreSQL) 7.4.6`	http://www.postgresql.org/
Eclipse	`Eclipse Platform Version: 3.0.1`	http://www.eclipse.org/

άλλων εκδόσεων των παρακάτω εργαλείων, και ειδικότερα προηγούμενων εκδόσεων, είναι πιθανόν να απαιτηθούν ορισμένες προσαρμογές στο σύστημα, αν και έχουμε κάνει κάθε προσπάθεια να μειώσουμε την εξάρτηση από ιδιαιτερότητες συγκεκριμένων των εκδόσεων.

5.2 Υλοποίηση των υποσυστημάτων

Στη συνέχεια περιγράφουμε συνοπτικά τον τρόπο υλοποίησης κάθε υποσυστήματος καθώς και τα ονόματα και τη λειτουργικότητα των βασικών κλάσεων του πηγαίου κώδικα. Ας σημειωθεί στο σημείο αυτό ότι ο πηγαίος κώδικας συνοδεύεται από πλήρη τεκμηρίωση, η οποία είναι διαθέσιμη σε ηλεκτρονική μορφή (JavaDoc) και συνοδεύει την παρούσα εργασία. Η τεκμηρίωση αυτή θα φανεί χρήσιμη σε οποιονδήποτε επιθυμεί να επεκτείνει το σύστημα αυτό.

5.2.1 Υποσύστημα διεπαφής με το δίκτυο ομότιμων κόμβων

Το υποσύστημα αυτό έχει υλοποιηθεί με τη βοήθεια της βιβλιοθήκης JXTA, η οποία περιγράφηκε στην Ενότητα 5.1.4. Στη συνέχεια θα δώσουμε ορισμένες λεπτομέρειες για τον τρόπο με τον οποίο χρησιμοποιήθηκε η συγκεκριμένη βιβλιοθήκη.

Πολλές λειτουργίες του υποσυστήματος αυτού υλοποιούνται στην κλάση `PeerManager`. Με την εκκίνηση του κόμβου γίνεται η αρχικοποίηση της πλατφόρμας JXTA ζητώντας να αποκτήσουμε πρόσβαση στο καθολικό σύνολο των κόμβων του δικτύου (Net Peer Group).

Στη συνέχεια οι κόμβοι του δικτύου μας ομαδοποιήθηκαν δημιουργώντας ένα μια 'ομάδα κόμβων' η οποία στην ορολογία του JXTA ονομάζεται Peer Group. Χρησιμοποιήθηκαν οι συναρτήσεις που μας παρέχει το API του JXTA για τη δημιουργία και την συμμετοχή στο Peer Group. Η λειτουργικότητα αυτή περιλαμβάνεται στην κλάση `PeerGroupUtils`.

Μόλις ο κόμβος γίνει μέλος του συγκεκριμένου Peer Group, είναι δυνατή η απαρίθμηση των υπόλοιπων μελών μέσω της υπηρεσίας `DiscoveryService` του JXTA. Η

λειτουργικότητα αυτή περιέχεται επίσης στην κλάση `PeerManager`.

Για την ανταλλαγή μηνυμάτων μεταξύ των κόμβων χρησιμοποιήθηκαν οι κλάσεις `JxtaServerSocket` και `JxtaSocket` που παρέχει το JXTA. Οι κλάσεις αυτές παρέχουν ένα API όμοιο με το API των 'sockets' που χρησιμοποιούνται για σύνδεση μεταξύ δύο μηχανημάτων πάνω από δίκτυα TCP/IP. Το API αυτό είναι γνωστό, καθιερωμένο και εύκολο στη χρήση. Οι βιβλιοθήκες του JXTA χειρίζονται όλες τις λεπτομέρειες της πραγματικής μετάδοσης των δεδομένων χωρίς την παρέμβαση του χρήστη.

Κάθε κόμβος που αναμένει συνδέσεις δημιουργεί ένα αντικείμενο `JxtaServerSocket`. Η έννοια της διεύθυνσης και πόρτας IP (IP Address/Port) που χρησιμοποιείται στα TCP/IP sockets αντικαθίσταται στα JXTA sockets από ένα αντικείμενο του JXTA που ονομάζεται Peer-ID (αναγνωριστικό κόμβου). Αυτό είναι μια μοναδική τυχαία δεκαεξαδική συμβολοσειρά η οποία στην περίπτωσή μας δημιουργείται με βάση το όνομα του κόμβου από την κλάση `MD5ID`. Το αναγνωριστικό αυτό ανακοινώνεται μέσω του JXTA σε όλο το δίκτυο έτσι ώστε να γίνει διαθέσιμο στους κόμβους που πιθανόν να θελήσουν να συνδεθούν. Η κλάση η οποία υλοποιεί την παραπάνω λειτουργικότητα είναι η `ConnectionServer`.

Κάθε κόμβος που θέλει να συνδεθεί με έναν άλλον δημιουργεί ένα αντικείμενο `JxtaSocket` με παράμετρο το αναγνωριστικό του απομακρυσμένου κόμβου. Στον απομακρυσμένο κόμβο, η κλάση `ConnectionServer` δημιουργεί ένα νέο αντικείμενο `JxtaSocket` γι'αυτό το άκρο της σύνδεσης, και ειδοποιεί έναν χειριστή συμβάντων (event handler) ο οποίο υλοποιεί το interface `ConnectionListener`. Στην εφαρμογή μας το ρόλο του χειριστή συμβάντων (καθώς και γενικότερα ρόλο συντονιστή του κόμβου) έχει η κλάση `PeerManager`. Αφού αποκατασταθεί η επικοινωνία μεταξύ των δύο κόμβων, ο `PeerManager` σε κάθε άκρο της σύνδεσης αρχικοποιεί μια κλάση που υλοποιεί το interface `ConnectionHandler` και οι οποία αναλαμβάνει στη συνέχει την αναγνώριση και το χειρισμό των μηνυμάτων.

5.2.2 Υποσύστημα διεπαφής με τη βάση RDF

Το υποσύστημα αυτό έχει υλοποιηθεί με τη βοήθεια του συστήματος RDFSuite, το οποίο περιγράφηκε στην Ενότητα 5.1.5.

Συγκεκριμένα, για την αποθήκευση σχημάτων στη βάση χρησιμοποιήθηκε το API που παρέχει το RSSDB, ενώ για την εκτέλεση ερωτημάτων το API του RQL Interpreter. Οι λειτουργίες αυτές έχουν ενσωματωθεί στην κλάση `RDFDatabaseManager`.

Μιας και το αποτέλεσμα της RQL είναι κι αυτό ένα αρχείο RDF, χρειάζεται να το διαβάσουμε για να εξάγουμε τα δεδομένα που μας ενδιαφέρουν σε κάποια δομή δεδομένων στη μνήμη. Το έργο αυτό επιτελεί η κλάση `RQLResultParser` η οποία με τη σειρά της χρησιμοποιεί το SAX API για να διαβάσει το αρχείο των αποτελεσμάτων το οποίο βρίσκεται σε μορφή XML. Έχει οριστεί επίσης η κλάση `RQLException` η οποία χρησιμοποιείται για να δηλώσει σφάλμα κατά την εκτέλεση κάποιου ερωτήματος RQL.

Τέλος, στην κλάση `RDFDatabaseManager` έχουν ενσωματωθεί συναρτήσεις που εκτελούν ορισμένα προκαθορισμένα ερωτήματα, τα οποία είναι χρήσιμα για το υποσύστημα

δημιουργίας τυχαίων σχημάτων.

5.2.3 Υποσύστημα λειτουργίας του κόμβου

Το υποσύστημα αυτό αλληλεπιδρά με τα υπόλοιπα υποσυστήματα με σκοπό το χειρισμό των μηνυμάτων και γενικά την εκτέλεση των λειτουργιών του κόμβου. Στο υποσύστημα ανήκουν οι παρακάτω κλάσεις:

- Η κλάση `XMLMsgConnectionHandler` η οποία υλοποιεί το interface `ConnectionHandler`. Για κάθε σύνδεση στον κόμβο, το υποσύστημα διεπαφής p2p δημιουργεί ένα στιγμιότυπο της κλάσης αυτής, η οποία αναλαμβάνει στη συνέχεια την αποστολή και λήψη δεδομένων. Τα δεδομένα που ανταλλάσσονται είναι κατάλληλα μορφοποιημένα μηνύματα, η μορφή των οποίων περιγράφεται στην Ενότητα 5.3.
- Η κλάση `PeerLogic` η οποία περιέχει όλη τη λογική της λειτουργίας του κόμβου ('business logic') όπως περιγράφηκε στην ενότητα 4.2. Στην κλάση αυτή υλοποιείται η 'συμπεριφορά' του κόμβου, δηλαδή οι λειτουργίες που θα εκτελεί, οι απαντήσεις που θα στέλνει και γενικά η αντίδρασή του σε κάθε μήνυμα.

5.2.4 Υποσύστημα δημιουργίας τυχαίων σχημάτων

Το συγκεκριμένο υποσύστημα βασίστηκε εν μέρει στον αλγόριθμο που υλοποιείται στο σύστημα RDFSculpt και περιγράφεται στο [KDS05]. Στη συνέχεια περιγράφεται ο αλγόριθμος αυτός μαζί με τις δικές μας επεκτάσεις.

Για την παράγωγη ενός τυχαίου σχήματος από ένα αρχικό εκτελούνται τα παρακάτω βήματα:

1. Επιλέγεται ένα αρχικό σχήμα RDF/S και αποθηκεύεται στη βάση RDF.
2. Απαριθμούνται οι κλάσεις του αρχικού σχήματος και επιλέγεται ένα τυχαίο υποσύνολό τους. Η επιλογή γίνεται ως εξής: Από το χρήστη καθορίζεται παράμετρος του αλγορίθμου την οποία ονομάζουμε πιθανότητα επιλογής κλάσεων (class selection probability), έστω p_C όπου $0 < p_C \leq 1$. Για κάθε μια από τις κλάσεις υπολογίζεται ένας τυχαίος αριθμός R όπου $0 \leq R \leq 1$. Μια κλάση επιλέγεται αν και μόνο αν $R < p_C$.
3. Για καθεμιά από τις επιλεγμένες κλάσεις, απαριθμούνται οι ιδιότητες του σχήματος που έχουν ως πεδίο ορισμού την κλάση αυτή. Από τις ιδιότητες αυτές επιλέγονται τυχαία ορισμένες, με βάση μια δεύτερη παράμετρο την οποία ονομάζουμε πιθανότητα επιλογής ιδιοτήτων (property selection probability).
4. Για καθεμιά από τις επιλεγμένες ιδιότητες:
   (a) Βρίσκεται ποιες από τις αρχικά επιλεγμένες κλάσεις ανήκουν στο ευρύτερο πεδίο ορισμού της ιδιότητας αυτής. Αν υπάρχουν παραπάνω από μια κλάσεις, αυτό σημαίνει ότι στο αρχικό σχήμα η ιδιότητα είχε οριστεί για μια κοινή υπερκλάση

(αφού δεν επιτρέπονται παραπάνω από μια κλάσεις στο πεδίο ορισμού). Η κοντινότερη κοινή υπερκλάση (nearest common ancestor) προστίθεται κι αυτή στις επιλεγμένες κλάσεις (εφ'όσον δεν ανήκει ήδη) και τοποθετείται ως πεδίο ορισμού της επιλεγμένης ιδιότητας.

(b) Βρίσκεται ποιες από τις αρχικά επιλεγμένες κλάσεις ανήκουν στο ευρύτερο πεδίο τιμών της ιδιότητας αυτής, εφ'όσον το πεδίο τιμών της κλάσης δεν είναι κάποιος τύπος λεκτικού (literal). Αν υπάρχουν παραπάνω από μια κλάσεις, αυτό σημαίνει ότι στο αρχικό σχήμα η ιδιότητα είχε πεδίο τιμών μια κοινή τους υπερκλάση (αφού δεν επιτρέπονται παραπάνω από μια κλάσεις στο πεδίο τιμών). Η κοντινότερη κοινή υπερκλάση (nearest common ancestor) προστίθεται κι αυτή στις επιλεγμένες κλάσεις (εφ'όσον δεν ανήκει ήδη) και τοποθετείται ως πεδίο τιμών της επιλεγμένης ιδιότητας.

Με επανάληψη όλων των παραπάνω βημάτων εκτός από το πρώτο μπορούμε να πάρουμε πολλά διαφορετικά σχήματα, τα οποία θα αποτελούν όλα έγκυρα υποσύνολα του αρχικού σχήματος, κατά την έννοια του ορισμού 3.2.

Στη συνέχεια με τυχαίο τρόπο προσθέτουμε σε κάθε σχήμα τα παρακάτω:

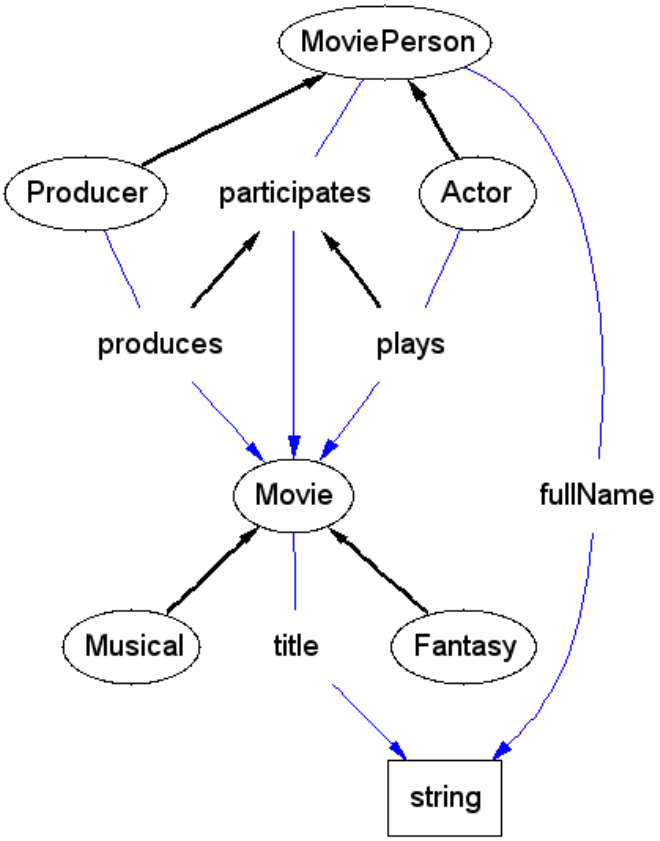
- Νέες κλάσεις οι οποίες αποτελούν υποκλάσεις των επιλεγμένων σε κάθε σχήμα κλάσεων. Η επιλογή των υπερκλάσων και των ονομάτων για τις νέες κλάσεις γίνεται τυχαία. Ομοίως επιλέγεται τυχαία (μέσα σε κάποια όρια που ορίζονται από το χρήστη) ο αριθμός των νέων κλάσεων που προσθέτονται.

- Νέες ιδιότητες οι οποίες έχουν πεδίο ορισμού τις επιλεγμένες σε κάθε σχήμα κλάσεων. Η επιλογή των κλάσεων πεδίου ορισμού, των ονομάτων και των πεδίων τιμών των νέων ιδιοτήτων γίνεται τυχαία. Ομοίως επιλέγεται τυχαία (μέσα σε κάποια όρια που ορίζονται από το χρήστη) ο αριθμός των νέων ιδιοτήτων που προσθέτονται. Οι νέες ιδιότητες έχουν πάντα πεδίο τιμών κάποιον τύπο λεκτικού.

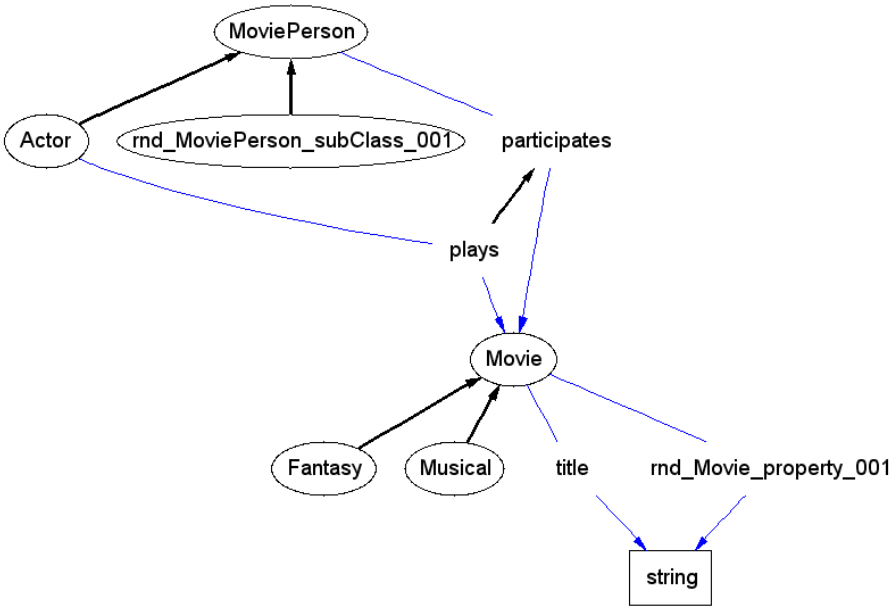
Οι παραπάνω προσθήκες γίνονται με τέτοιο τρόπο ώστε να μην προκύψουν συγκρούσεις μεταξύ των παραγόμενων RDF σχημάτων, τα δε σχήματα έχουν όπως προαναφέραμε ένα κοινό υποσύνολο άρα είναι συμβατά μεταξύ τους κατά την έννοια του ορισμού 3.3.

Ένα παράδειγμα αρχικού σχήματος που μπορεί να χρησιμοποιηθεί ως είσοδος στη γεννήτρια τυχαίων σχημάτων φαίνεται στο Σχήμα 5.2. Ένα τυχαίο σχήμα που παρήγαγε η γεννήτρια τυχαίων σχημάτων φαίνεται στο Σχήμα 5.3. Παρατηρούμε ότι στο δεύτερο σχήμα υπάρχουν μόνο ορισμένες από τις κλάσεις και τις ιδιότητες του αρχικού σχήματος. Ταυτόχρονα, έχουν προστεθεί διάφορες τυχαίες κλάσεις και ιδιότητες, οι οποίες ξεχωρίζουν από το πρόθεμα `rnd_` που έχουν στο όνομά τους.

Η κλάση που υλοποιεί την παραπάνω λειτουργικότητα είναι η `RDFSchemaGenerator`. Έχει δημιουργηθεί επίσης η κλάση `RDFSFileWriter` η οποία γράφει τα δημιουργούμενα σχήματα σε μορφή αρχείου RDF/XML.



Σχήμα 5.2: Αρχικό σχήμα για τη γέννηση τυχαίων σχημάτων



Σχήμα 5.3: Παράγωγο σχήμα από τη γέννηση τυχαίων σχημάτων

5.2.5 Υποσύστημα δημιουργίας τυχαίων τοπολογιών

Όπως αναφέρθηκε στην Ενότητα 4.1.5, η τοπολογία του συστήματός μας θα έχει τη μορφή ενός κατευθυνόμενου άκυκλου γράφου (DAG). Ένας τέτοιος γράφος με 10 κόμβους φαίνεται στο Σχήμα 5.4. Στη συνέχεια θα περιγράψουμε τον αλγόριθμο ο οποίος χρησιμοποιείται για να δημιουργεί τέτοιους τυχαίους γράφους.

Ως γνωστόν ένας κατευθυνόμενος γράφος με N κόμβους μπορεί να περιγραφεί με έναν πίνακα E διαστάσεων $N \times N$ ο οποίος ονομάζεται πίνακας γειτονίας (adjacency matrix). Κάθε στοιχείο e_{ij} του πίνακά αυτού μπορεί να παίρνει μια από τις τιμές $(0, 1)$. Ισχύει $e_{ij} = 1$ αν και μόνο αν στο γράφο υπάρχει ακμή με κατεύθυνση από τον i-οστό προς τον j-οστό κόμβο, ειδάλλως $e_{ij} = 0$. Αποδεικνύεται ότι, εάν $e_{ij} = 0 \forall (i, j) : i \leq j$ τότε ο κατευθυνόμενος γράφος που περιγράφεται από τον πίνακα E είναι ταυτόχρονα άκυκλος. Στη συνέχεια θα ασχοληθούμε με γράφους που πληρούν το κριτήριο αυτό. Για παράδειγμα, ο πίνακας γειτονίας για το γράφο του Σχήματος 5.4 είναι ο εξής:

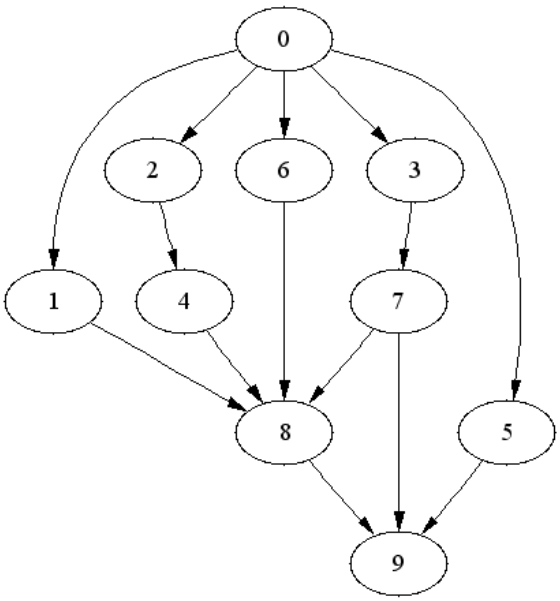
$$E = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Ο αλγόριθμος παραγωγής τυχαίων γράφων ξεκινά κατασκευάζοντας έναν πίνακα E στον οποίο $e_{ij} = 1 \forall (i, j) : i > j$, δηλαδή υπάρχουν όλες οι επιτρεπόμενες ακμές μεταξύ των κόμβων ώστε ο γράφος να παραμένει άκυκλος σύμφωνα με το προαναφερθέν κριτήριο. Ο αριθμός των ακμών αυτών είναι $N_e = \frac{N(N-1)}{2}$. Η παράμετρος του αλγορίθμου που καθορίζεται από το χρήστη, έστω p όπου $p_{min} < p \leq 1$, ονομάζεται connection factor και αφορά το ποσοστό των παραπάνω συνδέσεων οι οποίες θα παραμείνουν στον τελικό γράφο. Στους γράφους που παράγονται από τον αλγόριθμο πρέπει να συμμετέχουν όλοι οι κόμβοι, άρα απαιτούνται τουλάχιστον $N - 1$ ακμές. Έτσι, το p είναι κάτω φραγμένο: $p \times N_e \geq N - 1 \Rightarrow p_{min} = \frac{2}{N}$.

Στη συνέχεια ο αλγόριθμος προχωρά διαγράφοντας από το γράφο $N_d = (1 - p)N_e$ ακμές με τυχαίο τρόπο. Δεν επιτρέπεται να διαγραφεί μια ακμή εάν μετά τη διαγραφή κάποιος από τους κόμβους θα παραμείνει χωρίς καμιά εισερχόμενη η εξερχόμενη ακμή. Έτσι εξασφαλίζεται η συμμετοχή όλων τον κόμβων στον τελικό γράφο.

Ο συγκεκριμένος αλγόριθμος σε συνδυασμό με τον περιορισμό που θέσαμε για τους γράφους δημιουργεί DAGs στους οποίους:

- Υπάρχει πάντα ένας μόνο κόμβος (ο κόμβος 0) από τον οποίον μόνο ξεκινούν ακμές.



Σχήμα 5.4: Ένας κατευθυνόμενος άκυκλος γράφος (DAG)

Αυτό είναι μια επιθυμητή ιδιότητα και μας επιτρέπει να έχουμε σε όλες τις παραγόμενες τοπολογίες ένα γνωστό σημείο για την τοποθέτηση του επιβλέποντα κόμβου.

- Υπάρχει πάντα ένας μόνο κόμβος (ο κόμβος $N-1$) στον οποίο μόνο καταλήγουν ακμές. Σε περίπτωση που αυτό δεν είναι επιθυμητό,μπορεί να δημιουργηθεί ένας γράφος με $N+1$ κόμβους από τον οποίον να αφαιρέσουμε τον τελευταίο κόμβο.

Η υλοποίηση του παραπάνω αλγορίθμου καθώς και της αναπαράστασης του γράφου με πίνακα γειτονίας περιέχονται στην κλάση `DAG`.

5.2.6 Υποσύστημα δημιουργίας τυχαίων ερωτημάτων

Το υποσύστημα λειτουργίας τυχαίων ερωτημάτων βασίζεται σε μια λίστα από πρότυπα ερωτήματα (query patterns). Καθένα από τα πρότυπα αυτά ερωτήματα μπορεί να περιέχει διάφορα πεδία (arguments) τα οποία αντικαθιστώνται με τυχαία δεδομένα κατά το χρόνο εκτέλεσης. Παραδείγματα πρότυπων ερωτημάτων φαίνονται στο Σχήμα 5.5.

```
Class
subClassOf({0})
{0} in subPropertyOf({1})
nca({0}, {1})
SELECT @P FROM {;{0}}@P
```

Σχήμα 5.5: Πρότυπα ερωτήματα

Τα πεδία εμφανίζονται μέσα στα πρότυπα με τη χρήση της ειδικής ακολουθίας {x}, όπου

x ο αύξων αριθμός του πεδίου. Κάθε πρότυπο συνοδεύεται από μια λίστα που καθορίζει τον τύπο των πεδίων, δηλαδή το είδος των δεδομένων με τα οποία θα αντικατασταθούν. Για παράδειγμα, το πρότυπο `nca({0}, {1})` μπορεί συνοδεύεται από τη λίστα `('c', 'c')`. Η λίστα αυτή υποδηλώνει ότι τόσο το πρώτο όσο και το δεύτερο πεδίο θα αντικατασταθούν με ονόματα τυχαίων κλάσεων. Γενικά, με τον χαρακτήρα `'c'` υποδηλώνουμε μια τυχαία κλάση, ενώ με το χαρακτήρα `'p'`.

Στο Σχήμα 5.6 φαίνονται ορισμένα τυχαία ερωτήματα που δημιουργήθηκαν με βάση κάποια πρότυπα όπως τα παραπάνω.

```
Fantasy in subClassOf(MoviePerson)
subPropertyOf(generates)
Class
subClassOf(Producer)
Fantasy in subClassOf(Producer)
```

Σχήμα 5.6: Τυχαία παραγόμενα ερωτήματα

5.2.7 Υποσύστημα συλλογής μετρήσεων

Το υποσύστημα αυτό κάνει δυο ειδών μετρήσεις:

- Μέτρηση χρόνου μετάδοσης μηνυμάτων. Για να γίνει η μέτρηση αυτή, ο επιβλέπων κόμβος συνδέεται με έναν άλλον κόμβο και στέλνει δοκιμαστικό μήνυμα με κενό περιεχόμενο (ping). Ο δεύτερος κόμβος στη συνέχει απαντά με άλλο ένα δοκιμαστικό μήνυμα (ping reply). Η μέτρηση αφορά το χρόνο από την αρχή την αποστολή του πρώτου μηνύματος μέχρι το τέλος της λήψης του δεύτερου μηνύματος (Round-Trip-Time, RTT).

- Μέτρηση χρόνου εκτέλεσης ερωτημάτων. Για να γίνει η μέτρηση αυτή, κάθε κόμβος που απαντά ένα ερώτημα μετρά το χρόνο εκτέλεσης του ερωτήματος (από τη στιγμή που θα κληθεί μέχρι τη στιγμή που θα απαντήσει ο διερμηνέας της RQL). Στη συνέχεια, εκτός από τα αποτελέσματα του ερωτήματος επιστρέφει στον αρχικό κόμβο ένα δεύτερο μήνυμα στο οποίο αναφέρει το ερώτημα που εκτελέστηκε και το χρόνο που χρειάστηκε για την εκτέλεσή του.

Τα αποτελέσματα των παραπάνω μετρήσεων γράφονται σε ένα αρχείο εξόδου σε μορφή εύκολα αναγνώσιμη και επεξεργάσιμη από άλλες εφαρμογές. Η μορφή των εγγραφών για τις δύο παραπάνω μετρήσεις φαίνεται αντίστοιχα στα Σχήματα 5.7 και 5.8. Στην πρώτη περίπτωση η εγγραφή αποτελείται από τη λέξη `ping`, τα ονόματα των δύο κόμβων και το χρόνο σε milliseconds. Στη δεύτερη περίπτωση, ξεκινά με τη λέξη `query`, το όνομα του κόμβου που εκτέλεσε το ερώτημα και το χρόνο εκτέλεσης σε milliseconds. Στη συνέχεια παρατίθεται το ερώτημα RQL και η εγγραφή τερματίζεται με τη λέξη `query-end`.

```
ping peer-0 peer-4 431
```

Σχήμα 5.7: Εγγραφή μέτρησης χρόνου μετάδοσης (ping)

```
query peer-7 210
subClassOf(Artist)
query-end
```

Σχήμα 5.8: Εγγραφή μέτρησης χρόνου εκτέλεσης ερωτήματος

5.2.8 Υποσύστημα διεπαφής χρήστη

Η αλληλεπίδραση με το χρήστη γίνεται σε δύο επίπεδα. Το ένα από αυτά είναι τα μηνύματα που παρουσιάζονται σε μορφή κειμένου στο κέλυφος (shell) του Unix. Για το σκοπό αυτό χρησιμοποιείται η βιβλιοθήκη καταγραφής μηνυμάτων (logging) Log4J η οποία παρέχει έναν ευέλικτο τρόπο παρακολούθησης της λειτουργίας του συστήματος. Μια ευκολία του Log4J είναι η αυτόματη καταγραφή του σημείου στον πηγαίο κώδικα (με λεπτομέρεια κλάσης, συνάρτησης και αριθμού γραμμής) από το οποίο προέκυψε κάθε μήνυμα. Έτσι διευκολύνεται ο εντοπισμός και η διόρθωση τυχόν προβλημάτων. Επίσης, το Log4J μπορεί να ομαδοποιήσει τα μηνύματα σε ορισμένες κατηγορίες ανάλογα με τη σοβαρότητά τους και να παρουσιάσει μόνον αυτά που ζητά ο χρήστης. Μπορούμε π.χ. να παρακολουθήσουμε με μεγάλη λεπτομέρεια την εσωτερική λειτουργία του προγράμματος ή να επιλέξουμε να εμφανίζονται μηνύματα μόνο σε περιπτώσεις σοβαρών σφαλμάτων που απαιτούν προσοχή. Ας σημειωθεί επίσης ότι το Log4J χρησιμοποιείται ήδη από την πλατφόρμα JXTA για την εμφάνιση των δικών της μηνυμάτων, οπότε όλα τα μηνύματα του προγράμματός μας εμφανίζονται με ενιαίο τρόπο.

Ειδικά για τον επιβλέποντα κόμβο έχει αναπτυχθεί και γραφικό περιβάλλον (GUI) το οποίο επιτρέπει την παρακολούθηση της λειτουργίας του συστήματος καθώς και την εκτέλεση των χειρισμών που ορίστηκαν στην περιγραφή του υποσυστήματος αυτού. Το γραφικό αυτό περιβάλλον έχει υλοποιηθεί με τη βοήθεια της βιβλιοθήκης Swing η οποία βρίσκεται ανάμεσα στις βασικές βιβλιοθήκες της γλώσσας Javakαι παρέχει όλα όσα χρειαζόμαστε για ένα πλήρες γραφικό περιβάλλον. Η κύρια κλάση που αφορά το γραφικό περιβάλλον είναι η `PeerManagerUI`.

5.2.9 Υποσύστημα εγκατάστασης του δικτύου

Το υποσύστημα αυτό αποτελείται από ένα σύνολο προγραμμάτων φλοιού του Unix (shell scripts) τα οποία φροντίζουν για την ορθή εκκίνηση ενός αριθμού κόμβων. Τα προγράμματα αυτά χειρίζονται ορισμένες λεπτομέρειες σχετικές με το JXTA. Σε κάθε κόμβο δίνεται ένα μοναδικό όνομα της μορφής `peer-X` όπου `X` ο αύξων αριθμός του κόμβου. Επίσης, καθορίζεται διαφορετικός φάκελος εκκίνησης (current directory) για καθέναν από τους κόμβους που τρέχουν στο ίδιο μηχάνημα, έτσι ώστε κάθε κόμβος να έχει τον ιδιωτικό του χώρο

αποθήκευσης των στοιχείων που χρειάζονται για τη λειτουργία του. Τέλος, δημιουργείται το κατάλληλο αρχείο ρυθμίσεων για κάθε κόμβο.

Το βασικό πρόγραμμα που εκτελεί τις παραπάνω εργασίες είναι το `runPeer.sh` το οποίο εκτελείται με παράμετρο τον αύξοντα αριθμό του κόμβου.

Για τη μαζική εκκίνηση ενός συνόλου κόμβων, χρησιμοποιείται το πρόγραμμα `runNetwork.sh` το οποίο εκτελείται με παράμετρο τον αριθμό των κόμβων που θέλουμε να εκκινησουμε.

5.3 Επικοινωνία μεταξύ των κόμβων

Όπως αναφέρθηκε στην ενότητα 5.2.1 οι κόμβοι επικοινωνούν μεταξύ τους ανοίγοντας αμφίδρομες συνδέσεις με τη μορφή 'sockets'. Στη συνέχεια θα περιγράψουμε τη μορφή των μηνυμάτων που ανταλλάσσουν οι κόμβοι μεταξύ τους, τον τρόπο μετάδοσής τους και τη σημασία τους.

5.3.1 Περιεχόμενο και μετάδοση των μηνυμάτων

Κάθε μήνυμα αποτελείται από από ένα τμήμα που αποτελεί το ωφέλιμο περιεχόμενό του (payload) καθώς και από ορισμένες επικεφαλίδες (headers). Οι επικεφαλίδες είναι οι παρακάτω:

- Πηγή μηνύματος (message source) : το όνομα του κόμβου ο οποίος μεταδίδει το μήνυμα.
- Πηγή αναμεταδιδόμενου μηνύματος (propagate message source) : το όνομα του κόμβου από τον οποίο ξεκίνησε η μετάδοση ενός αναμεταδιδόμενου μηνύματος (για την έννοια των αναμεταδιδόμενων μηνυμάτων βλ. παρακάτω).
- Τύπος μηνύματος (message type) : ένα αναγνωριστικό που δηλώνει τη λειτουργία του συγκεκριμένου μηνύματος και βοηθά τον παραλήπτη να ερμηνεύσει το ωφέλιμο περιεχόμενο.

Τα μηνύματα μεταδίδονται μεταξύ των κόμβων σε μορφή XML ανάλογη με αυτήν που φαίνεται στο Σχήμα 5.9. Οι επικεφαλίδες του μηνυματος μεταδίδονται ως ιδιότητες (attributes) της ετικέτας `message`. Το ωφέλιμο περιεχόμενο του μηνύματός συμπιέζεται και κωδικοποιείται πριν από τη μετάδοση σε μορφή Base64[1]. Η κωδικοποίηση Base64 μετατρέπει μια σειρά από bytes σε μια άλλη σειρά από bytes που αποτελείται μόνο από εκτυπώσιμους χαρακτήρες. Με τον τρόπο αυτό εξασφαλίζουμε ότι μπορεί να μεταδοθεί οποιοδήποτε περιεχόμενο χωρίς να δημιουργηθούν προβλήματα στον XML parser που διαβάζει το μήνυμα. Επιπλέον, η συμπίεση GZIP που χρησιμοποιείται μειώνει το μέγεθος του μηνύματος, ειδικά όταν μεταδίδουμε μεγάλα RDF αρχεία τα οποία μπορούν να συμπιεστούν σε μεγάλο βαθμό.

Όσον αφορά τον τρόπο μετάδοσης των μηνυμάτων, μπορούμε να διακρίνουμε τρεις τύπους μετάδοσης:

[1] βλ. RFC-2045, http://www.ietf.org/rfc/rfc2045.txt

```
<?xml version="1.0"?>
<message source="sourcePeerName"
         propagateSource="propagateSourcePeerName"
         type="messageType">

... message payload ...

</message>
```

Σχήμα 5.9: Η μορφή XML των μηνυμάτων

- Σύγχρονη μετάδοση μηνύματος (synchronous message transfer), όπου ένας κόμβος μεταδίδει ένα μήνυμα σε έναν άλλον και περιμένει (μέσω της ίδιας σύνδεσης) ένα άλλο μήνυμα ως απάντηση.

- Ασύγχρονη μετάδοση μηνύματος (asynchronous message transfer), όπου ένας κόμβος μεταδίδει ένα μήνυμα σε έναν άλλον και στη συνέχεια κλείνει τη σύνδεση χωρίς να αναμείνει απάντηση. Εφ'όσον ο δεύτερος κόμβος θέλει να επιστρέψει κάποια δεδομένα, πρέπει ο ίδιος να συνδεθεί εκ νέου στον πρώτο κόμβο.

- Αναμετάδοση μηνύματος (propagate message transfer), όπου ένας κόμβος μεταδίδει ένα μήνυμα σε όλους τους γείτονές του, οι οποίοι με τη σειρά τους το μεταδίδουν στους δικούς τους γείτονες κ.ο.κ. (flooding) σύμφωνα με τα στοιχεία τοπολογίας που έχουν οριστεί σε κάθε κόμβο. Εφ'όσον κάποιοι από τους κόμβους θέλουν να επιστρέψουν κάποια δεδομένα, πρέπει οι ίδιοι να συνδεθούν άμεσα στην πηγή του μηνύματος. Όλα τα μηνύματα στην περίπτωση αυτή μεταδίδονται ασύγχρονα.

5.3.2 Τύποι μηνυμάτων

Παρακάτω περιγράφονται τα διαφόρων ειδών μηνύματα που ανταλλάσσουν οι κόμβοι και το νόημα του ωφέλιμου περιεχομένου σε περίπτωση.

Μηνύματα για τον ορισμό τοπολογίας

Τα μηνύματα αυτά μεταδίδονται από τον επιβλέποντα κόμβο προς όλους τους υπόλοιπους κόμβους για τον ορισμό τον γειτόνων κάθε κόμβου. Ο επιβλέπων κόμβος συνδέεται με κάθε κόμβο και μεταδίδει ένα μήνυμα με τύπο `set-adjacent`. Το ωφέλιμο φορτίο έχει μορφή κειμένου και περιλαμβάνει μια λίστα με τα ονόματα των γειτόνων κόμβων όπου το κάθε όνομα βρίσκεται σε ξεχωριστή γραμμή. Η μετάδοση του μηνύματος είναι σύγχρονη και ο επιβλέπων κόμβος αναμένει ως απάντηση ένα μήνυμα με τύπο `set-adjacent-status`. Το περιεχόμενο της απάντησης είναι `OK` σε περίπτωση επιτυχημένης εκτέλεσης της λειτουργίας αυτής ή ένα μύνημα λάθους σε περίπτωση που παρουσιάστηκε κάποιο πρόβλημα.

Μια τυπική περίπτωση ανταλλαγής μηνυμάτων κατά τη διάρκεια του ορισμού τοπολογίας φαίνεται στο Σχήμα 5.10. Στο παράδειγμα αυτό ο κόμβος `peer-0` ενημερώνει τον κόμβο `peer-2` ότι γείτονές του είναι οι κόμβοι `peer-4`, `peer-3` και `peer-6`, και ο κόμβος `peer-2` απαντά ότι η λειτουργία εκτελέστηκε χωρίς πρόβλημα.

Σημειώνουμε ότι στην πραγματικότητα το ωφέλιμο φορτίο του μηνύματος μεταδίδεται κωδικοποιημένο όπως προαναφέραμε, αλλά στο παράδειγμα αυτό καθώς και στα επόμενα παραδείγματα το παρουσιάζουμε ως απλό κείμενο για λόγους επίδειξης.

```
> <?xml version="1.0"?>
> <message source="peer-0"
>          type="set-adjacent">
> peer-4
> peer-3
> peer-6
> </message>

< <?xml version="1.0"?>
< <message source="peer-2"
<         type="set-adjacent-status">
< OK
< </message>
```

Σχήμα 5.10: Ανταλλαγή μηνυμάτων κατά τον ορισμό τοπολογίας

Μηνύματα για τον ορισμό σχημάτων

Τα μηνυματα αυτά μεταδίδονται από τον επιβλέποντα κόμβο προς όλους τους υπόλοιπους κόμβους για τον ορισμό του σχήματος που χρησιμοποιεί κάθε κόμβος. Ο επιβλέπων κόμβος συνδέεται με κάθε κόμβο και μεταδίδει ένα μήνυμα με τύπο `store-rdfs`. Το ωφέλιμο φορτίο έχει μορφή RDF/XML και περιλαμβάνει το σχήμα προς αποθήκευση. Η μετάδοση του μηνύματος είναι σύγχρονη και ο επιβλέπων κόμβος αναμένει ως απάντηση ένα μήνυμα με τύπο `store-rdfs-status`. Το περιεχόμενο της απάντησης είναι `OK` σε περίπτωση επιτυχημένης εκτέλεσης της λειτουργίας αυτής ή ένα μήνυμα λάθους σε περίπτωση που παρουσιάστηκε κάποιο πρόβλημα.

Μια τυπική περίπτωση ανταλλαγής μηνυμάτων κατά τη διάρκεια του ορισμού σχημάτων φαίνεται στο Σχήμα 5.11. Στο παράδειγμα αυτό ο κόμβος `peer-0` στέλνει στον κόμβο `peer-2` το σχήμα RDF/S που αποτελεί το ωφέλιμο φορτίο του μηνύματος, και ο κόμβος `peer-2` απαντά ότι η λειτουργία εκτελέστηκε χωρίς πρόβλημα.

```
> <?xml version="1.0"?>
> <message source="peer-0"
>          type="store-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#"
>         xml:base="http://futureshape.net/rdfp2p/peer2">
>
><rdfs:Class rdf:ID="Movie"/>
>
>                 .........
></rdf:RDF>
> </message>

< <?xml version="1.0"?>
< <message source="peer-2"
<         type="store-rdfs-status">
< OK
< </message>
```

Σχήμα 5.11: Ανταλλαγή μηνυμάτων κατά τον ορισμό σχημάτων

Μηνύματα για την εκτέλεση ερωτημάτων

Τα μηνύματα αυτά μεταδίδονται από τον επιβλέποντα κόμβο ως μηνύματα αναμετάδοσης (propagate) για την εκτέλεση ερωτημάτων RQL. Ο επιβλέπων κόμβος συνδέεται με όλους τους γείτονες του μεταδίδει στον καθένα ένα μήνυμα με τύπο `rql-query`. Το μήνυμα αυτό αναμεταδίδεται στο δίκτυο μέσω του μηχανισμού αναμετάδοσης μηνυμάτων και φτάνει τελικά σε όλους τους κόμβους. Το ωφέλιμο φορτίο έχει μορφή κειμένου και περιλαμβάνει ένα ερώτημα γραμμένο σύμφωνα με το συντακτικό της RQL (βλ. Ενότητα 2.1.2). Τα αποτελέσματα των ερωτημάτων μεταδίδονται μέσω μηνυμάτων τύπου `rql-query-results` τα οποία περιέχουν την απάντηση σε μορφή RDF/XML όπως επιστρέφεται από το διερμηνέα της RQL, καθώς και τα στατιστικά στοιχεία που επιστρέφονται μέσω μηνυμάτων τύπου `rql-query-time`.

Μια τυπική περίπτωση ανταλλαγής μηνυμάτων κατά τη διάρκεια της εκτέλεσης των ερωτημάτων φαίνεται στο Σχήμα 5.12. Στο παράδειγμα αυτό ο κόμβος `peer-4` στέλνει στον κόμβο `peer-2` το ερώτημα RQL που αποτελεί το ωφέλιμο φορτίο του μηνύματος. Στο μήνυμα αυτό υποδεικνύεται ότι αρχική πηγή του ερωτήματος ήταν ο κόμβος `peer-0`. Έτσι, ο κόμβος `peer-2` συνδέεται στη συνέχει με τον κόμβο `peer-0` και μεταδίδει τα αποτελέσματα

του ερωτήματος, καθώς και τα στατιστικά στοιχεία.

```
> <?xml version="1.0"?>
> <message source="peer-4"
>          type="rql-query"
>          propagateSource="peer-0">
> subClassOf(Artist)
> </message>

< <?xml version="1.0"?>
< <message source="peer-2"
<          type="rql-query-results">
< <?xml version="1.0" encoding="UTF-8"?>
< <RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
< <rdf:Bag>
< <rdf:li rdf:type="class" rdf:resource="Painter"/>
< <rdf:li rdf:type="class" rdf:resource="Flemish"/>
< <rdf:li rdf:type="class" rdf:resource="Cubist"/>
< <rdf:li rdf:type="class" rdf:resource="Sculptor"/>
< </rdf:Bag>
< </RDF>
<
< </message>

< <?xml version="1.0"?>
< <message source="peer-2"
<          type="rql-query-time-253">
<  subClassOf(Artist)
< </message>
```

Σχήμα 5.12: Ανταλλαγή μηνυμάτων κατά την εκτέλεση ερωτημάτων

Δοκιμαστικά μηνύματα

Τα μηνύματα αυτά μεταδίδονται από τον επιβλέποντα κόμβο για τη μέτρηση του χρόνου μετάδοσης (ping). Πρόκειται για ένα ζεύγος μηνυμάτων `ping` και `ping-reply`, τα οποία δεν περιέχουν κανένα ωφέλιμο φορτίο.

Μια τυπική περίπτωση ανταλλαγής δοκιμαστικών μηνυμάτων κατά τη διάρκεια της εκτέλεσης των ερωτημάτων φαίνεται στο Σχήμα 5.13. Στο παράδειγμα τα μηνύματα ανταλλάσσονται μεταξύ των κόμβων `peer-0` και `peer-2`.

```
> <?xml version="1.0"?>
> <message source="peer-0"
>          type="ping">
> </message>

< <?xml version="1.0"?>
< <message source="peer-2"
<          type="ping-reply">
< </message>
```

Σχήμα 5.13: Ανταλλαγή δοκιμαστικών μηνυμάτων

Κεφάλαιο 6

Έλεγχος

6.1 Μεθοδολογία ελέγχου

Ο έλεγχος της ορθής λειτουργίας του συστήματός μας έγινε σε δύο επίπεδα:

Αρχικά ελέγχθηκε ξεχωριστά κάθε υποσύστημα (unit testing). Για καθένα από τα υποσυστήματα δημιουργήθηκε ένα περιβάλλον ελέγχου (testbench). Στο περιβάλλον αυτό το υποσύστημα θεωρείται ότι είναι ένα 'μαύρο κουτί' (black box) με εισόδους και εξόδους που καθορίζονται από τη δημόσια διεπαφή του (public API). Στις εισόδους του υποσυστήματος δίνονται διάφορα τυχαία δεδομένα, καθώς και δεδομένα τα οποία περιέχουν οριακές τιμές, και πιστοποιείται η ορθότητα των δεδομένων εξόδου.

Είναι σημαντικό να σημειώσουμε ότι η διαδικασία αυτή δεν διεξάγεται στο τέλος της ανάπτυξης του συστήματος, αλλά παράλληλα με αυτήν. Η τακτική εκτέλεση των παραπάνω δοκιμών βοηθά να αποκαλυφθούν άμεσα τυχόν προβλήματα που προκύπτουν από αλλαγές σε ένα υποσύστημα.

Σε δεύτερο επίπεδο, και αφού έχει προχωρήσει η ανάπτυξη του συστήματος δοκιμάζεται η συνεργασία μεταξύ των υποσυστημάτων, μέσω διαφόρων σεναρίων λειτουργίας του συστήματος. Στην περίπτωση της δικιάς μας εργασίας, το στάδιο αυτό περιλαμβάνει και την δοκιμή της ορθής συνεργασίας των κόμβων του δικτύου για την εκτέλεση της λειτουργίας του συστήματός μας.

6.2 Έλεγχος υποσυστημάτων

6.2.1 Υποσύστημα δημιουργίας τυχαίων σχημάτων

Το υποσύστημα αυτό δοκιμάστηκε με διάφορα αρχικά σχήματα RDF/S ως είσοδο. Το υποσύστημα παρήγαγε πολλά τυχαία σχήματα βασισμένα σε κάποιο από τα αρχικά σχήματα. Όλα τα παραγόμενα σχήματα δοκιμάστηκαν με τους παρακάτω τρόπους:

- Επιβεβαίωση ότι το παραγόμενο σχήμα είναι ένα έγκυρο αρχείο XML (XML validation). Για την επιβεβαίωση αυτή χρησιμοποιήθηκε ένας συντακτικός αναλυτής XML (XML parser).

- Επιβεβαίωση ότι το παραγόμενο σχήμα είναι ένα έγκυρο σχήμα RDF/S αναπαριστάμενο σε RDF/XML (schema validation). Για την επιβεβαίωση αυτή χρησιμοποιήθηκε ο συντακτικός αναλυτής VRP (Validating RDF Parser) ο οποίος ανήκει στο σύστημα RDFSuite.

6.2.2 Υποσύστημα δημιουργίας τυχαίων τοπολογιών

Το υποσύστημα αυτό δοκιμάστηκε με διαφορετικές τιμές για τον αριθμό των κόμβων και το ποσοστό των συνδέσεων (connection factor). Για καθέναν από αυτούς του συνδυασμούς τιμών επιβεβαιώθηκε ότι:

- Ο παραγόμενος γράφος είναι ένας έγκυρος κατευθυνόμενος άκυκλος γράφος.
- Στον παραγόμενο γράφο συμμετέχουν τόσοι κόμβοι όσοι προδιαγράψαμε.

Οι έλεγχοι αυτοί υλοποιήθηκαν με έναν ξεχωριστό αλγόριθμο.

6.2.3 Υποσύστημα δημιουργίας τυχαίων ερωτήσεων

Το υποσύστημα αυτό δοκιμάστηκε με διάφορα αρχικά σχήματα RDF/S ως είσοδο. Το υποσύστημα παρήγαγε πολλές τυχαίες ερωτήσεις για καθένα από τα αρχικά σχήματα. Οι παραγόμενες ερωτήσεις επιβεβαιώθηκαν ότι είναι συντακτικά ορθές χρησιμοποιώντας το διερμηνέα της RQL.

6.2.4 Υποσύστημα διεπαφής με τη βάση RDF

Για τον έλεγχο του υποσυστήματος αυτού χρησιμοποιήθηκαν τα αποτελέσματα από τα υποσυστήματα δημιουργίας τυχαίων σχημάτων και ερωτήσεων. Δημιουργήθηκαν διάφορα τυχαία σχήματα τα οποία και αποθηκεύτηκαν στη βάση RDF. Για καθένα από αυτά τα σχήματα δημιουργήθηκαν οι ανάλογες ερωτήσεις οι οποίες εστάλησαν προς απάντηση. Επιβεβαιώθηκε ότι:

- Η διασύνδεση με τη βάση δεδομένων λειτουργεί σωστά.
- Δεν παρουσιάζονται σφάλματα κατά την αποθήκευση των σχημάτων και την απάντηση ερωτήσεων.
- Το αποθηκευμένο σχήμα συμφωνεί με την περιγραφή του σε RDF/XML.

6.2.5 Υποσύστημα διεπαφής με το δίκτυο p2p

Για τον έλεγχο του υποσυστήματος αυτού χρησιμοποιήθηκαν τα αποτελέσματα από το υποσύστημα δημιουργίας τυχαίων τοπολογιών. Δημιουργήθηκε διάφορα τυχαία δίκτυα και πάνω στα δίκτυα αυτά μεταδόθηκαν δοκιμαστικά μηνύματα ('ping' messages). Επιβεβαιώθηκε ότι:

- Όλοι οι κόμβοι μπορούν να επικοινωνήσουν μεταξύ τους.

- Τα αναμεταδιδόμενα μηνύματα ακολουθούν την πορεία που περιγράφεται από την τοπολογία του δικτύου.

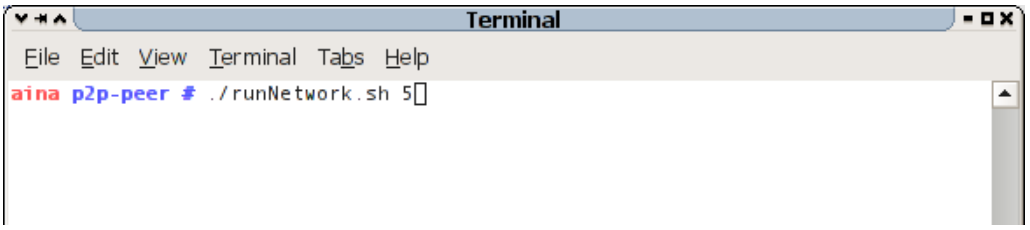
6.3 Έλεγχος ολοκληρωμένου συστήματος

Για τον έλεγχο του ολοκληρωμένου συστήματος χρησιμοποιήσαμε ένα τυπικό σενάριο εξομοίωσης το οποίο και εκτελέσαμε πολλές φορές για να διαπιστώσουμε εάν παρουσιάζονται προβλήματα.

Αν και το σενάριο αυτό μπορεί να εκτελεστεί αυτόματα, εμείς θα παρουσιάσουμε στην συνέχεια την χειροκίνητη εκτέλεσή του, περνώντας από όλα τα ενδιάμεσα βήματα και δείχνοντας ταυτόχρονα τον τρόπο λειτουργίας του συστήματος.

6.3.1 Εκκίνηση του συστήματος

Για την εκκίνηση του συστήματος πρέπει ο τρέχων φάκελος να είναι αυτός στον οποίο έχει εγκατασταθεί το σύστημα. Από τον φάκελο αυτό εκτελούμε το πρόγραμμα φλοιού (shell script) `runNetwork.sh` με παράμετρο τον αριθμό των κόμβων που θέλουμε να εκκινήσουμε. Στο παράδειγμά μας θα εκκινήσουμε ένα δίκτυο με 5 κόμβους (βλ. Σχήμα 6.1).



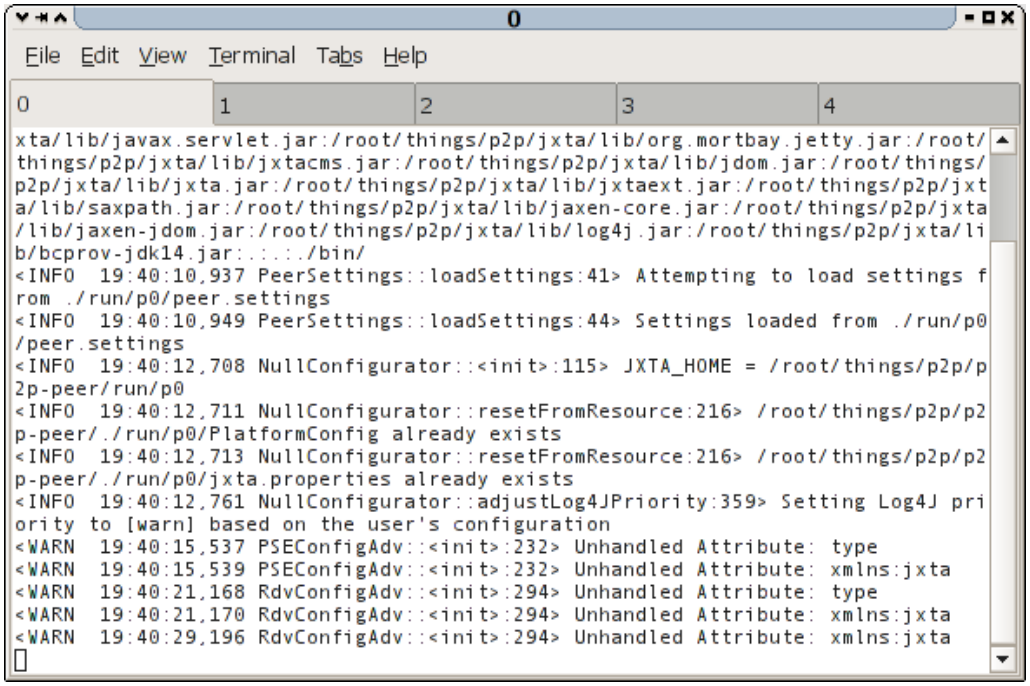
Σχήμα 6.1: Εκκίνηση του συστήματος

Στη συνέχεια θα εμφανιστεί ένα νέο παράθυρο τερματικού με πολλές σελίδες, στην καθεμιά από τις οποίες εμφανίζονται τα μηνύματα που γράφει κάθε κόμβος στη γραμμή εντολών (standard output). Το παράθυρο αυτό απεικονίζεται στο Σχήμα 6.2.

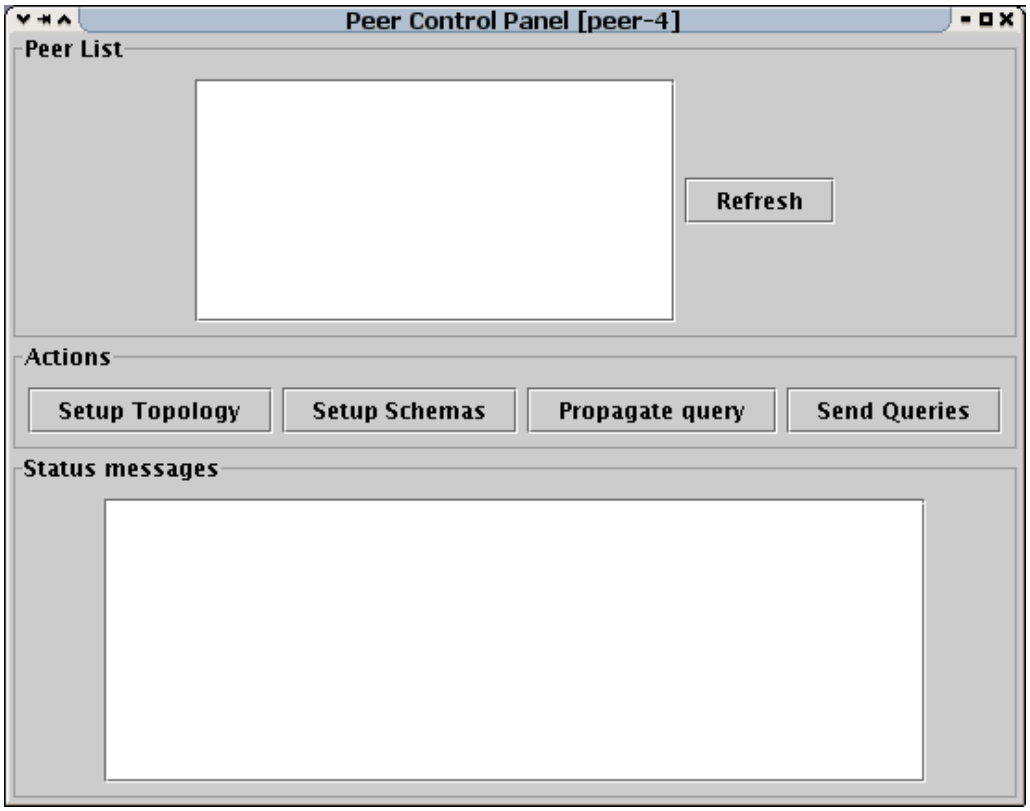
Παράλληλα, θα εμφανιστεί η γραφική διεπαφή χρήστη (GUI) για κάθε κόμβο. Στο Σχήμα 6.3 μπορούμε να δούμε μια απεικόνιση του GUI της εφαρμογής. Παρατηρούμε ότι είναι χωρισμένο σε τρία μέρη. Στο πρώτο μέρος φαίνεται μια λίστα με τους υπόλοιπους κόμβους του δικτύου (Peer List). Στο δεύτερο μέρος υπάρχουν ορισμένα κουμπιά τα οποία ενεργοποιούν διάφορες λειτουργίες της εφαρμογής (Actions). Τέλος, το τρίτο μέρος είναι αφιερωμένο στην εμφάνιση μηνυμάτων.

6.3.2 Απαρίθμηση των κόμβων του δικτύου

Για να προχωρήσουν οι υπόλοιπες λειτουργίες, πρέπει πρώτα να αναζητηθούν οι υπόλοιποι κόμβοι που συμμετέχουν στο δίκτυο, καθώς το δίκτυο διαμορφώνεται δυναμικά και κάθε κόμβος αρχικά δε γνωρίζει τίποτα για τους υπόλοιπους. Επιλέγοντας το κουμπί 'Refresh' (Ανανέωση) ενημερώνεται η λίστα με τους κόμβους που είναι ενεργοί στο δίκτυό μας. Μετά

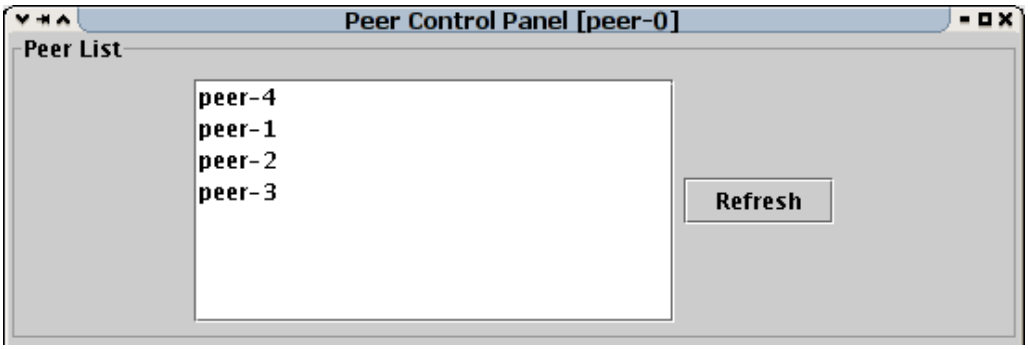


Σχήμα 6.2: Παράθυρο μηνυμάτων



Σχήμα 6.3: Η γραφική διεπαφή χρήστη του συστήματος

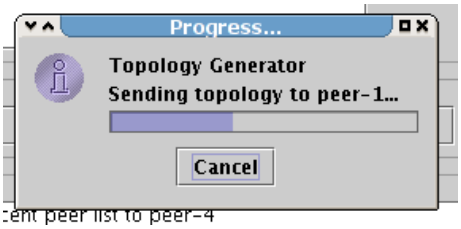
την ολοκλήρωση αυτής της λειτουργίας, στη λίστα θα πρέπει να έχουν προστεθεί ορισμένα ονόματα κόμβων, όπως στο Σχήμα 6.4.



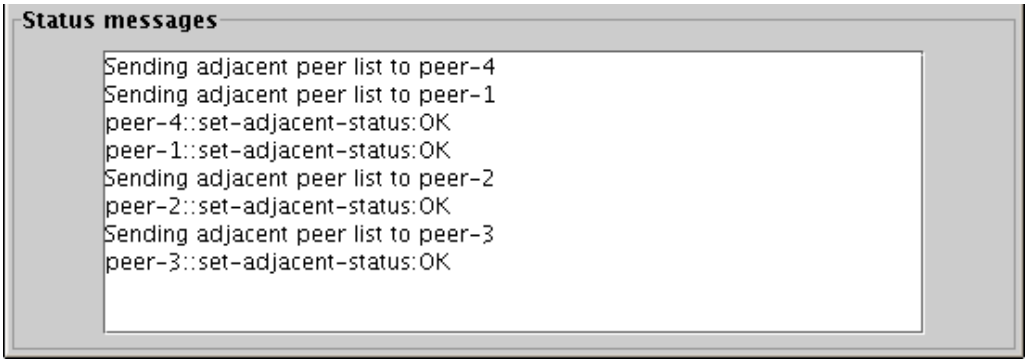
Σχήμα 6.4: Λίστα κόμβων μετά την ανανέωση

6.3.3 Εγκατάσταση τοπολογιών

Πρώτο μας μέλημα είναι να εγκαταστήσουμε στο δίκτυο μια τυχαία τοπολογία. Η διαδικασία αυτή ενεργοποιείται πατώντας το κουμπί 'Setup Topology' (Εγκατάσταση Τοπολογίας). Θα παρουσιαστεί ένα παράθυρο όμοιο με αυτό του Σχήματος 6.5 που θα μας ενημερώνει για την πρόοδο της εργασίας. Αφού ολοκληρωθεί η εργασία, θα εμφανιστούν στην περιοχή μηνυμάτων τα αντίστοιχα μηνύματα που επιβεβαιώνουν την ορθή εκτέλεσή της ή ενημερώνουν για τυχόν σφάλματα (βλ. Σχήμα 6.6).



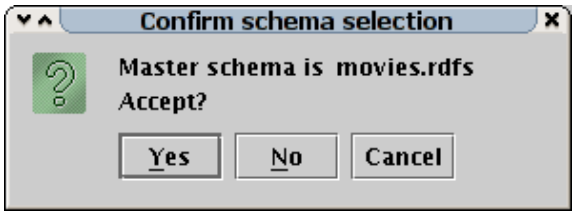
Σχήμα 6.5: Πρόοδος της εγκατάστασης τοπολογίας



Σχήμα 6.6: Μηνύματα μετά την εγκατάσταση τοπολογίας

6.3.4 Εγκατάσταση σχημάτων

Στη συνέχεια, πρέπει να σταλούν τυχαία σχήματα RDF/S σε κάθε κόμβο. Η διαδικασία αυτή ενεργοποιείται πατώντας το κουμπί 'Setup Schemas' (Εγκατάσταση Σχημάτων). Θα εμφανιστεί ένα παράθυρο διαλόγου όμοιο με αυτό του Σχήματος 6.7 το οποίο ζητά να επιβεβαιώσουμε εάν θέλουμε να χρησιμοποιήσουμε το προεπιλεγμένο σχήμα. Εάν επιλέξουμε 'Yes' (Ναι), θα προχωρήσει η διαδικασία με το προεπιλεγμένο σχήμα. Εάν επιλέξουμε 'No' ('Οχι), θα εμφανιστεί ένας επιλογέας αρχείων (file chooser) ώστε να επιλέξουμε κάποιο άλλο αρχείο. Τέλος, εάν επιλέξουμε 'Cancel' (Ακύρωση), η διαδικασία δεν θα εκκινήσει.

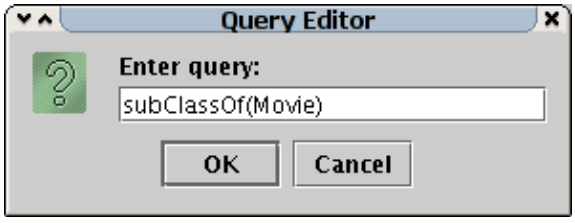


Σχήμα 6.7: Επιλογή αρχικού σχήματος

Κατά τη διάρκεια της διαδικασίας εγκατάστασης σχημάτων θα εμφανιστούν, όπως και στη διαδικασία εγκατάστασης τοπολογίας, ένα παράθυρο προόδου και τα αντίστοιχα μηνύματα.

6.3.5 Αποστολή ερωτημάτων

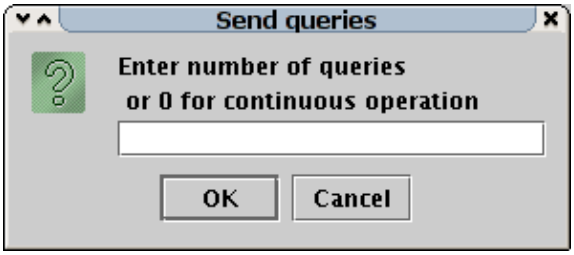
Αφού τα δύο προηγούμενα βήματα ολοκληρωθούν με επιτυχία, το δίκτυο μπορεί να χρησιμοποιηθεί για την απάντηση ερωτημάτων. Για την αποστολή ενός μόνο ερωτήματος πρέπει να πατηθεί το κουμπί 'Propagate Query' (αναμετάδοση ερωτήματος). Θα εμφανιστεί ένα παράθυρο διαλόγου όμοιο με αυτό του Σχήματος 6.8 όπου ζητείται να εισάγουμε το ερώτημα σε μορφή RQL.



Σχήμα 6.8: Εισαγωγή ερωτήματος

Για τη συνεχή αποστολή τυχαίων ερωτημάτων πρέπει να πατηθεί το κουμπί 'Send Queries' (αποστολή ερωτημάτων). Θα εμφανιστεί ένα παράθυρο διαλόγου όμοιο με αυτό του Σχήματος 6.9 όπου ζητείται να εισάγουμε τον αριθμό των ερωτημάτων που θέλουμε να στείλουμε. Εάν εισάγουμε τον αριθμό '0', το σύστημα θα συνεχίσει να στέλνει τυχαία ερωτήματα έως ότου τερματίσουμε τη διαδικασία.

Κατά τη διάρκεια της διαδικασίας εμφανίζεται ένα παράθυρο προόδου που παρουσιάζει το ποσοστό των ερωτημάτων που έχουν σταλεί μέχρι στιγμής και δίνει την δυνατότητα



Σχήμα 6.9: Επιλογή αριθμού ερωτημάτων

τερματισμού της διαδικασίας. Τα αποτελέσματα των ερωτημάτων σε μορφή RDF/XML εμφανίζονται παράλληλα στην περιοχή μηνυμάτων της εφαρμογής.

Κεφάλαιο 7

Επίλογος

7.1 Αποτελέσματα και συμπεράσματα

Τελικό αποτέλεσμα της παρούσας εργασίας είναι ένας πλήρως λειτουργικός εξομοιωτής, έτοιμος να χρησιμοποιηθεί για έρευνα στα δίκτυα ομότιμων κόμβων με σχήματα. Κατά την άποψή μας, το όραμα ενός παγκόσμιου δικτύου p2p που θα παρέχει πληροφορία (σε αντίθεση με τα σημερινά δίκτυα που παρέχουν απλώς δεδομένα) είναι εφικτό, εφ'όσον λυθούν ορισμένα ανοιχτά ερευνητικά προβλήματα. Ευελπιστούμε πως το σύστημά μας θα βοηθήσει την επίτευξη του στόχου αυτού.

Το σύστημά μας φτιάχτηκε με στόχο να είναι όσο το δυνατόν πιο εύχρηστο για τους τελικούς χρήστες του, δηλαδή τους ερευνητές που θα το χρησιμοποιήσουν. Πιστεύουμε πως ο στόχος αυτός έχει επιτευχθεί, καθώς:

- έχουμε χωρίσει το σύστημα σε απλά, διακριτά υποσυστήματα τα οποία μπορούν εύκολα να τροποποιηθούν ή να αντικατασταθούν για να καλύψουν τις ανάγκες εξομοίωσης μιας οποιασδήποτε σχεδίασης δικτύου.

- έχουμε επιλέξει τη χρήση ανοιχτών και προτυποποιημένων τεχνολογιών όπως το JXTA και το RDF, με αποτέλεσμα να εξασφαλίζεται η αλληλεπίδραση με άλλα συστήματα που χρησιμοποιούν τα πρότυπο αυτά.

- έχουμε παράγει διεξοδική τεκμηρίωση για τη σχεδίαση και λειτουργία του συστήματος, τόσο σε υψηλό επίπεδο (ανάλυση και σχεδίαση) όσο και σε χαμηλό επίπεδο (προγραμματιστικές διεπαφές, APIs).

- έχουμε φροντίσει ώστε η εγκατάστασή και η ρύθμιση του συστήματος να γίνεται κατά το δυνατόν αυτόματα, με την ελάχιστη δυνατή ανθρώπινη παρέμβαση.

Ο πηγαίος κώδικας και η τεκμηρίωση που τον συνοδεύει θα διατεθούν ελεύθερα σε κάθε ενδιαφερόμενο μέσω του διαδικτύου. Σκοπός μας είναι η τεχνογνωσία που αποκτήθηκε κατά την εκπόνηση της διπλωματικής αυτής να φανεί χρήσιμη στην κοινότητα των χρηστών και προγραμματιστών που αναπτύσσουν δίκτυα ομότιμων κόμβων με το JXTA ή πειραματίζονται με το πλαίσιο RDF.

7.2 Μελλοντικές επεκτάσεις

Οι άξονες πάνω στους οποίους μπορεί να επεκταθεί το σύστημά μας είναι δύο.

Ο πρώτος άξονας αφορά την επαναϋλοποίηση ενός η περισσότερων υποσυστημάτων με σκοπό την υποστήριξη διαφορετικών προτύπων και τεχνολογιών. Ενδεικτικά αναφέρουμε τα εξής:

- Υλοποίηση του υποσυστήματος διεπαφής p2p με βάση κάποια άλλη πλατφόρμα δικτύου ομότιμων κόμβων.
- Υλοποίηση του υποσυστήματος διεπαφής RDF με βάση έναν διαφορετικό τρόπο αποθήκευσης δεδομένων RDF ή/και μια διαφορετική γλώσσα διατύπωσης ερωτημάτων.
- Υλοποίηση των υποσυστημάτων δημιουργίας τοπολογιών, σχημάτων και ερωτημάτων με διαφορετικούς περιορισμούς για τον τύπο των δεδομένων που παράγουν.

Ο δεύτερος άξονας αφορά την υλοποίηση νέων υποσυστημάτων με σκοπό την υποστήριξη και την εξομοίωση μιας πραγματικής ή ερευνητικής εφαρμογής. Ενδεικτικά αναφέρουμε τα εξής:

- Ανάπτυξη αλγορίθμων απάντησης ερωτημάτων που συνδυάζουν δεδομένα από πολλούς κόμβους με διαφορετικά σχήματα.
- Ανάπτυξη αλγορίθμων τροποποίησης ερωτημάτων (query-rewriting) με βάση δηλωμένες αντιστοιχίες ανά ζεύγος κόμβων.

Όλες οι παραπάνω τροποποιήσεις μπορούν να γίνουν στο σύστημά μας με άμεσο τρόπο, αφού μελετηθεί η σχετική τεκμηρίωση.

Βιβλιογραφία

[ACK+01] Sofia Alexaki, Vassilis Christophides, Gregory Karvounarakis, Dimitris Plexousakis, and Karsten Tolle. The ICS-FORTH RDFSuite: Managing Voluminous RDF Description Bases. In *SemWeb*, 2001.

[BG04] Dan Brickley and Ramanathan V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, 2004.

[BLHL01] T. Berners-Lee, J. Hendler, and O. Lassila. Semantic web. *Scientific American*, 284(5):34–43, 2001.

[Bra04] Marshall Brain. How Gnutella Works. HowStuffWorks.com, 2004.

[Gon01] Li Gong. JXTA: A Network Programming Environment. *IEEE Internet Computing*, 5(3):88–95, 2001.

[HIMT03] Alon Y. Halevy, Zachary G. Ives, Peter Mork, and Igor Tatarinov. Piazza: data management infrastructure for semantic web applications. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 556–567, New York, NY, USA, 2003. ACM Press.

[KC04] Giorgos Kokkinidis and Vassilis Christophides. Semantic Query Routing and Processing in P2P Database Systems: The ICS FORTH SQPeer Middleware. In *HDMS*, 2004.

[KDS05] Zoi Kaoudi, Theodore Dalamagas, and Timos K. Sellis. RDFSculpt: Managing RDF Schemas Under Set-Like Semantics. In *ESWC*, pages 123–137, 2005.

[MM04] Frank Manola and Eric Miller. RDF Primer, W3C Recommendation, 2004.

[NWQ+02] Wolfgang Nejdl, Boris Wolf, Changtao Qu, Stefan Decker, Michael Sintek, Ambjörn Naeve, Mikael Nilsson, Matthias Palmér, and Tore Risch. EDUTELLA: a P2P networking infrastructure based on RDF. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pages 604–615, New York, NY, USA, 2002. ACM Press.

[Tys04] Jeff Tyson. How the Old Napster Worked. HowStuffWorks.com, 2004.

[Zhu05] Hai Zhuge. The future interconnection environment. *IEEE Computer*, 38(4):29, April 2005.