Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Πληροφορική»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Υπηρεσία Αυτόματης Ανάκτησης Συνδεδεμένης Δομής Θεματικών Επικεφαλίδων μέσω της Βιβλιοθήκης του Κογκρέσου** Service Autorecover Structure Linked Subject Headings through the Library of Congress
Ονοματεπώνυμο Φοιτητή	**Καλλιόπη Καναβού**
Πατρώνυμο	**Μιλτιάδης**
Αριθμός Μητρώου	**ΜΠΠΛ/ 10048**
Επιβλέπων	**Ιωάννης Παπαδάκης, Επ. Καθηγητής**

Ημερομηνία Παράδοσης **Μήνας Έτος**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Χρήστος Δουληγέρης Καθηγητής	Δέσποινα Πολέμη Αναπληρώτρια Καθηγήτρια	Δημήτριος Βέργαδος Επίκουρος Καθηγητής

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΑΦΙΕΡΩΣΗ

Η παρούσα αυτή μεταπτυχιακή διατριβή είναι αφιερωμένη στον αγαπημένο μου Γιώργο Καραπάνο.

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα εργασία αποτελεί το τελικό λιθαράκι για την κτήση του μεταπτυχιακού διπλώματος στον τομέα της Πληροφορικής του τμήματος “Πληροφορική” του Πανεπιστημίου Πειραιώς.

Αισθάνομαι την ανάγκη να ευχαριστήσω τον καθηγητή μου κ. Ιωάννη Παπαδάκη, για την εμπιστοσύνη που μου έδειξε, την υπομονή του και για όλες εκείνες τις ατελείωτες ώρες συλλογικής δουλειάς στα εργαστήρια του τμήματος.

Θέλω επίσης να ευχαριστήσω όλους τους καθηγητές του τμήματος για τις γνώσεις που μου παρείχαν. Δεν μπορώ να παραλείψω τους συμφοιτητές μου Χρήστο Γιαννούλη και Δημήτρη Κουγιουμτζή για την άψογη συνεργασία στις ομαδικές εργασίες κατά τη διάρκεια σπουδών του μεταπτυχιακού προγράμματος.

Τέλος να ευχαριστήσω την οικογένειά μου, τους γονείς μου Μίλτο και Γεωργία και τον αδερφό μου Κωνσταντίνο, για την συμπαράσταση, την στήριξη και την υπομονή κατά τα έτη σπουδών, αλλά και κατά τη διάρκεια της υλοποίησης και της συγγραφής της παρούσας μεταπτυχιακής διατριβής.

ΠΕΡΙΛΗΨΗ

Το θέμα της παρούσας μεταπτυχιακής διατριβής αναφέρεται σε μια Υπηρεσία Διαχείρισης των Θεματικών Επικεφαλίδων του online Καταλόγου (Online Public Access Catalog – OPAC) μιας Βιβλιοθήκης, που δημιουργήθηκε για να εξυπηρετήσει τις ανάγκες του σύγχρονου βιβλιοθηκονόμου.

Η Υπηρεσία αυτή δίνει τη δυνατότητα στον χρήστη να διαχειρίζεται τις θεματικές επικεφαλίδες μιας βιβλιοθήκης σε πραγματικό χρόνο (on-line) με τη χρήση τεχνολογιών του σημασιολογικού ιστού, οι οποίες είναι αποθηκευμένες σε μια εξειδικευμένη βάση δεδομένων (triple store).

Στόχος για κάθε μία θεματική επικεφαλίδα της βιβλιοθήκης, είναι να εντοπιστούν οι μη καταγεγραμμένες σχέσεις της θεματικής επικεφαλίδας αυτής με τις υπόλοιπες που βρίσκονται στη βιβλιοθήκη. Προκειμένου να επιτευχθεί αυτό χρησιμοποιήθηκε το αποθετήριο των θεματικών επικεφαλίδων της βιβλιοθήκης του Κογκρέσου.

Η υπηρεσία που κατασκευάστηκε στο πλαίσιο της διπλωματικής εργασίας χρησιμοποίησε δεδομένα της βιβλιοθήκης του Ιονίου Πανεπιστημίου.

Λέξεις Κλειδιά: Ανοικτά Διασυνδεδεμένα Δεδομένα, Σημασιολογικός Ιστός, Ενιαίο Αναγνωριστικό Πόρου, 4Store, Γλώσσα Sparql, python, εξυπηρετητής twisted.

ABSTRACT

The topic of this master thesis refers to a Service Management subject headings of online Catalog (Online Public Access Catalog - OPAC) of the Library, created to serve the needs of librarian.

This Service enables the user to manage the subject headings in real time (on-line) with the use of Semantic Web technologies, which are stored in a specialized database (triple store).

Aim for each subject heading of the library is to identify unregistered relations thematic title of this with the rest of the library. To achieve this used repository of subject headings of the Library of Congress.

The service, which is built in this master thesis used data library of the Ionian University.

Key Words: Open Linked Data, Semantic Web, Uniform Resource Identifier, 4 Store, Sparql Query Language, Python Language, Twisted Server.

ΕΙΣΑΓΩΓΗ

Ο ρόλος του Επιστήμονα της Πληροφόρησης έχει αναπροσαρμοστεί λόγω της εξέλιξης της τεχνολογίας, αυξάνοντας συνεχώς τις απαιτήσεις των χρηστών. Οι παραδοσιακοί κατάλογοι των βιβλιοθηκών έχουν αντικατασταθεί από αυτοματοποιημένες υπηρεσίες που παρέχουν πρόσβαση στους απλούς χρήστες μέσω διαδικτύου. Οι βιβλιοθήκες παρέχουν ταυτόχρονα μεγάλο αριθμό υπηρεσιών συνδυάζοντας πολλαπλές μορφές πληροφορίας, οι οποίες βρίσκονται σε απομακρυσμένες βάσεις δεδομένων[1].

Παραδοσιακά, οι βιβλιοθήκες ανά τον κόσμο χρησιμοποιούν ως ελεγχόμενο λεξιλόγιο τις θεματικές επικεφαλίδες της βιβλιοθήκης του Κογκρέσου (LCSH – Library of Congress Subject Headings) οι οποίες διατίθενται μέσω τεχνολογιών του σημασιολογικού ιστού και των συνδεδεμένων δεδομένων (Linked Data) ειδικότερα. Απαραίτητη προϋπόθεση για τη συμμετοχή στο κίνημα των συνδεδεμένων δεδομένων είναι για κάθε μία θεματική επικεφαλίδα να παρέχεται ένας μόνιμος προσδιοριστής (URI – Uniform Resource Identifier). Οι θεματικές επικεφαλίδες σχετίζονται μεταξύ τους μέσω της συνδετικής δομής. Η σύνδεση αυτή των θεματικών επικεφαλίδων της βιβλιοθήκης του Κογκρέσου παρέχεται σε μηχαναγνώσιμη μορφή μέσω της διασύνδεσης αυτών των μόνιμων προσδιοριστών (Παπαδάκης, Κυπριανός, Στεφανιδάκης, & Μαυροπόδη, 2009).

Στην παρούσα διπλωματική εργασία, δημιουργήθηκε μια υπηρεσία που στόχο έχει να καταγράψει τη συνδετική δομή της βιβλιοθήκης του Ιονίου Πανεπιστημίου, χρησιμοποιώντας πληροφορίες από την αντίστοιχη υπηρεσία της βιβλιοθήκης του Κογκρέσου. Βασίζεται σε τεχνολογίες των ανοικτών συνδεδεμένων δεδομένων και του παγκόσμιου ιστού γενικότερα. Πιο συγκεκριμένα, για κάθε μία θεματική επικεφαλίδα της βιβλιοθήκης του Ιονίου Πανεπιστημίου, θέλουμε να εντοπιστούν οι μη καταγεγραμμένες σχέσεις της θεματικής επικεφαλίδας με τις υπόλοιπες που βρίσκονται στη βιβλιοθήκη αυτή. Έτσι λοιπόν, χρησιμοποιήθηκαν οι θεματικές επικεφαλίδες της βιβλιοθήκης του Κογκρέσου για να ανακαλυφθούν οι μη καταγεγραμμένες σχέσεις που συνιστούν τη συνδετική δομή των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου, με τη χρήση της υπηρεσίας ανοικτών δεδομένων θεματικών επικεφαλίδων της βιβλιοθήκης του Κογκρέσου.

Όπως προαναφέρθηκε, οι θεματικές επικεφαλίδες της βιβλιοθήκης του Κογκρέσου αντιστοιχούν σε μόνιμους προσδιοριστές που συνδέονται μεταξύ τους μέσω της συνδετικής δομής. Η συνδετική δομή εκφράζεται με το πρότυπο της απλής οργάνωσης της γνώσης (Skos - Simple Knowledge Organization System). Αντιθέτως οι θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου οι οποίες αντιστοιχούν σε έναν μόνιμο προσδιοριστή πόρου δεν συνδέονται μεταξύ τους μέσω συνδετικής δομής. Στόχος της εργασίας είναι η ανακάλυψη της συνδετικής δομής των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου χρησιμοποιώντας τη συνδετική δομή των θεματικών επικεφαλίδων της βιβλιοθήκης του Κογκρέσου. Για το σκοπό αυτό κατασκευάστηκε μια υπηρεσία βασισμένη σε τεχνολογίες του παγκόσμιου ιστού.

Η υπηρεσία αυτόματης ανάκτησης συνδεδεμένης δομής στηρίζεται στην αρχιτεκτονική τριών στρωμάτων (3 - Tier) επιτρέποντας επικοινωνία μεταξύ των εξυπηρετητών στα τρία αυτά επίπεδα.

Στην επόμενη ενότητα γίνεται αναφορά των τεχνολογιών του σημασιολογικού ιστού που χρησιμοποιήθηκαν για την μοντελοποίηση των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου προκειμένου να εντοπιστεί η συνδετική δομή των ανοικτών συνδεδεμένων δεδομένων της κάθε μίας θεματικής επικεφαλίδας[2].

---

[1] Πρόσβαση: http://www.ionio.gr/~toraki/infotech_met/infotech_met0708_ergasies/10_roloi_dais_papadopoulos.pdf Ημ. Πρόσβασης: 3/06/2014

[2] Πρόσβαση: http://linkeddatabook.com/editions/1.0/ Ημ. Πρόσβασης 10/6/2014

ΑΝΟΙΚΤΑ ΣΥΝΔΕΔΕΜΕΝΑ ΔΕΔΟΜΕΝΑ

Σε αυτή την ενότητα περιγράφεται η αρχιτεκτονική των ανοικτών συνδεδεμένων δεδομένων. Ο όρος συνδεδεμένα δεδομένα αναφέρεται σε ένα σύνολο βέλτιστων πρακτικών για τη δημοσίευση και τη διασύνδεση δομημένων δεδομένων στον παγκόσμιο ιστό. Αυτές οι βέλτιστες πρακτικές εισήχθησαν από τον Tim Berners-Lee, εμπνευστή του παγκόσμιου ιστού και κατ' επέκταση του σημασιολογικού ιστού (Heath & Bizer, 2011). Τα συνδεδεμένα δεδομένα βασίζονται σε τέσσερις αρχές. Οι αρχές είναι οι παρακάτω:

- Χρησιμοποιήστε μόνιμους προσδιοριστές πόρων (URI) ως ονόματα για τα πράγματα
- Χρησιμοποιήστε το πρωτόκολλο μεταφοράς υπερκειμένου (HTTP – Hypertext Transfer Protocol) μόνιμων προσδιοριστών πόρου, έτσι ώστε να μπορούν να προσπελαστούν αυτά τα ονόματα
- Όταν κάποιος προσπελαύνει ένα μόνιμο προσδιοριστή πόρου, να παρέχονται χρήσιμες πληροφορίες, χρησιμοποιώντας πρότυπα όπως Sparql (SPARql Query Language for RDF)
- Να περιλαμβάνονται σύνδεσμοι προς άλλους μόνιμους προσδιοριστές πόρων, έτσι ώστε να μπορούν να ανακαλυφθούν περισσότερα πράγματα (Bizer, Heath, & Berners-Lee, 2009).

Η βιβλιοθήκη του Κογκρέσου παρέχει μια μοντέρνα υπηρεσία ανοικτών συνδεδεμένων δεδομένων που βασίζεται στις παραπάνω αρχές. Πιο συγκεκριμένα, η κάθε μία θεματική επικεφαλίδα της βιβλιοθήκης του Κογκρέσου αντιστοιχεί σε έναν μόνιμο προσδιοριστή πόρου (URI), ενώ μεταξύ τους όλες οι θεματικές επικεφαλίδες συνδέονται μέσω της συνδετικής δομής. Η συνδετική δομή εκφράζεται μέσω του προτύπου της απλής οργάνωσης της γνώσης Skos που περιγράφεται στην επόμενη ενότητα.

Στην παρούσα μεταπτυχιακή εργασία παρουσιάζεται μια υπηρεσία που, βασισμένη στις παραπάνω αρχές, διαχειρίζεται θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου μέσω της βιβλιοθήκης του Κογκρέσου. Οι θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου δεν αντιστοιχούν σε μόνιμους προσδιοριστές πόρων ούτε είναι καταγεγραμμένη η συνδετική δομή τους. Παρ' όλα αυτά οι περισσότερες θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου έχουν προκύψει από μετάφραση των θεματικών επικεφαλίδων της βιβλιοθήκης του Κογκρέσου. Προκειμένου να εκμεταλλευτούμε τις δυνατότητες που παρέχουν τα ανοικτά συνδεδεμένα δεδομένα αντιστοιχίζουμε κάθε μία θεματική επικεφαλίδα της βιβλιοθήκης του Ιονίου Πανεπιστημίου με την ισοδύναμή της στη βιβλιοθήκη του Κογκρέσου.

Στην επόμενη ενότητα περιγράφεται πως η βιβλιοθήκη του Κογκρέσου παρέχει τις θεματικές επικεφαλίδες σε μηχαναγνώσιμη μορφή ως ενιαία αναγνωριστικά πόρου.

ΟΙ ΘΕΜΑΤΙΚΕΣ ΕΠΙΚΕΦΑΛΙΔΕΣ ΤΗΣ ΒΙΒΛΙΟΘΗΚΗΣ ΤΟΥ ΚΟΚΓΡΕΣΟΥ ΩΣ ΕΝΙΑΙΑ ΑΝΑΓΝΩΡΙΣΤΙΚΑ ΠΟΡΟΥ URIS

Η βιβλιοθήκη του Κογκρέσου αρχικά σε μια πειραματική εφαρμογή παγκόσμιου ιστού 'lcsh.info' δημοσίευσε λεξιλόγια LoC σε πραγματικό χρόνο σύμφωνα με το πρότυπο Skos. Σήμερα έχει οργανωθεί και λειτουργεί σε μορφή on-line υπηρεσίας 'http://id.loc.gov/authorities/'. Αυτό σημαίνει ότι η κάθε μία θεματική επικεφαλίδα προσδιορίζεται από έναν μόνιμο προσδιοριστή ως αναγνωριστικό πόρου. Ο μόνιμος προσδιοριστής πόρου αναφέρεται στο απλό σύστημα οργάνωσης γνώσης των θεματικών επικεφαλίδων Skos, όπου συνδέει τους ευρύτερους (Broader - BT), στενότερους (Narrower - NT) και συναφείς (Related - RT) θεματικούς όρους[3].

Στο παραδοσιακό διαδίκτυο, τα ενιαία αναγνωριστικά πόρων μέσω των προγραμμάτων περιήγησης παραπέμπουν σε φιλικές προς τον χρήστη ιστοσελίδες. Στο νέο περιβάλλον που διαμορφώνεται μέσω του σημασιολογικού ιστού κεντρικό ρόλο διαδραματίζει το πλαίσιο περιγραφής πόρων (RDF – Resource Description Framework). Σύμφωνα με το πρότυπο αυτό δίνεται η δυνατότητα πρόσβασης στην πληροφορία όχι μόνο από ανθρώπους αλλά και από

[3] Πρόσβαση: http://ivan-herman.name/2009/05/01/library-of-congress-subject-headings-in-skos-on-line/ Ημ. Πρόσβασης: 20/6/2014

διαδικτυακές υπηρεσίες. Η βιβλιοθήκη του Κογκρέσου παρέχει μια υπηρεσία η οποία προσφέρει τις θεματικές επικεφαλίδες σε μορφή RDF/XML ή N-Triples (Herman, 2009).

Χρησιμοποιώντας τις παραπάνω πληροφορίες που προσφέρει η βιβλιοθήκη του Κογκρέσου, δημιουργήθηκε μία εξειδικεύμενη βάση δεδομένων με τις θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου ακολουθώντας τους απλούς κανόνες του προτύπου Skos μοντελοποιημένες σε μορφή N-Triples. Στην επόμενη ενότητα περιγράφεται η μοντελοποίηση αυτή.

ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΤΩΝ ΘΕΜΑΤΙΚΩΝ ΕΠΙΚΕΦΑΛΙΔΩΝ ΤΗΣ ΒΙΒΛΙΟΘΗΚΗΣ ΤΟΥ ΚΟΓΚΡΕΣΟΥ ΣΥΜΦΩΝΑ ΜΕ ΤΟ ΠΡΟΤΥΠΟ SKOS

Το μοντέλο δεδομένων Skos παρέχει ένα πρότυπο, για τη μεταφορά των υπαρχόντων συστημάτων οργάνωσης γνώσης στο σημασιολογικό ιστό. Το Skos παρέχει μια γλώσσα για την ανάπτυξη και κοινή χρήση νέων συστημάτων οργάνωσης γνώσης. Μπορεί να χρησιμοποιηθεί μόνο του ή σε συνδυασμό με τυπικές γλώσσες αναπαράστασης όπως τη γλώσσα οντολογίας του παγκοσμίου ιστού (OWL – Web Ontology Language).

Το πρότυπο Skos χρησιμοποιείται ως ένα σύστημα οργάνωσης γνώσης το οποίο εκφράζει τα αντίστοιχα δεδομένα σε μηχαναγνώσιμη μορφή. Στη συνέχεια τα δεδομένα αυτά μπορούν και ανταλλάσσονται μεταξύ των εφαρμογών και τέλος να δημοσιεύονται στο διαδίκτυο.

Σε αυτή την ενότητα περιγράφεται ο πυρήνας του μοντέλου Skos, πιο συγκεκριμένα τα χαρακτηριστικά εκείνα που απαιτούνται για να αντιπροσωπεύουν το μοντέλο αυτό.

Στο Skos, οι έννοιες οι οποίες αποτελούν το θεμελιώδες στοιχείο του λεξιλογίου του απλού αυτού μοντέλου, ταυτίζονται με ενιαία αναγνωριστικά πόρων, επισημαίνονται με λεξιλογικές συμβολοσειρές σε μία ή περισσότερες φυσικές γλώσσες, οι οποίες τεκμηριώνονται με διάφορους τύπους σημείωσης. Σημασιολογικά σχετίζονται μεταξύ τους σε άτυπες ιεραρχίες και τα δίκτυα σύνδεσης συγκεντρώνονται σε εννοιολογικά συστήματα (W3C Working Group Note, 2011).

Η υπηρεσία που κατασκευάστηκε στο πλαίσιο της διπλωματικής εργασίας αυτής επιτρέπει την αποτύπωση της συνδεδεμένης δομής των θεματικών επικεφαλίδων του δημοσίου καταλόγου της βιβλιοθήκης του Ιονίου Πανεπιστημίου σύμφωνα με το πρότυπο Skos.

Τα στοιχεία του μοντέλου δεδομένων Skos αποτελούνται από κλάσεις και ιδιότητες. Το απλό σύστημα οργάνωσης της γνώσης Skos εισάγει την κλάση Skos:Concept, η οποία επιβεβαιώνει ότι ένας δεδομένος πόρος είναι μια έννοια. Αυτό γίνεται είτε με τη δημιουργία (ή την επαναχρησιμοποίηση) ενός ενιαίου αναγνωριστικού πόρου (URI) για να προσδιορίσει μοναδικά την έννοια, είτε προβάλλοντας αυτή με RDF, χρησιμοποιώντας την ιδιότητα rdf:type, προσδιορίζοντας τον πόρο αυτό ως ενιαίο αναγνωριστικό πόρου τύπου Skos:Concept (Skos W3C Working Group, 2009).

Ο πόρος για τον οποίο γίνεται αναφορά, μπορεί να είναι μια οντότητα ή οτιδήποτε άλλο. Η κλάση Skos:Concept δεν είναι η μοναδική κλάση που διατίθενται στο απλό σύστημα οργάνωσης της γνώσης Skos. Υπάρχουν επίσης και άλλες κλάσεις υψηλού επιπέδου όπως:

- Skos:Collection είναι μια συλλογή των εννοιών. Η ταξινόμηση των συλλογών μπορεί να χρησιμοποιηθεί με συλλεκτικές σημασιολογικές σχεσιακές ιδιότητες (π.χ. Skos:narrower), όπου ορίζεται ένα σύνολο εννοιών κάτω από ένα μοντέλο από ετικέτες κόμβων
- Skos:CollectableProperty είναι μια ιδιότητα που μπορεί να χρησιμοποιηθεί σε μία Skos:Collection
- Skos:ConceptScheme είναι ένα σχήμα εννοιών, που προαιρετικά περιλαμβάνει δηλώσεις σχετικά με τις σημασιολογικές σχέσεις μεταξύ των εννοιών. Ενδεικτικά αναφέρονται: θησαυροί, συστήματα ταξινόμησης, κατάλογοι αντικείμενων κλάσης, ταξινομήσεων, ορολογίες, κλπ.
- Skos:OrderedCollection ορίζει μια διατεταγμένη συλλογή των εννοιών

Ο πυρήνας του Skos χρησιμοποιεί ιδιότητες αναθέτοντας μία ετικέτα σε ένα πόρο, όπου η ετικέτα αυτή υποδηλώνει την περιγραφή του πόρου σε μια φυσική γλώσσα. Οι ιδιότητες αυτές είναι οι Skos:altLabel και Skos:prefLabel[4].

Οι σημασιολογικές σχέσεις διαδραματίζουν καίριο ρόλο για τον καθορισμό των εννοιών. Το νόημα μιας έννοιας δεν ορίζεται μόνο από τις λέξεις φυσικής γλώσσας στις ετικέτες της, αλλά και από τους δεσμούς της με άλλες έννοιες στο λεξιλόγιο. Αντικατοπτρίζοντας τις βασικές κατηγορίες των σχέσεων που χρησιμοποιούνται στα λεξιλόγια, όπως οι θησαυροί, το μοντέλο Skos παρέχει τρεις τυπικές ιδιότητες:

Οι ιδιότητες ευρύτερος θεματικός όρος (Skos:broader) και στενότερος θεματικός όρος (Skos:narrower) επιτρέπουν την εκπροσώπηση των ιεραρχικών δεσμών. Στα πλαίσια της εργασίας, χρησιμοποιούμε τις δύο αυτές ιδιότητες προκειμένου να καταγραφούν οι ιεραρχικές σχέσεις μεταξύ των θεματικών επικεφαλίδων του Ιονίου Πανεπιστημίου.

Η ιδιότητα σχετικός θεματικός όρος (Skos:related) επιτρέπει την εκπροσώπηση των συνειρμικών (μη ιεραρχικών) σχέσεων. Στα πλαίσια της εργασίας, χρησιμοποιούμε την ιδιότητα αυτή προκειμένου να καταγραφούν οι μη ιεραρχικές σχέσεις μεταξύ των θεματικών επικεφαλίδων του Ιονίου Πανεπιστημίου.

Στην παρούσα μεταπτυχιακή διατριβή, οι έννοιες εκφράζουν τις θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου οι οποίες επισημαίνονται με ετικέτες χαρακτήρων μιας συμβολοσειράς (Unicode), σε δύο φυσικές γλώσσες, Αγγλικά και Ελληνικά. Μια από αυτές τις ετικέτες μεταξύ των δύο γλωσσών μπορεί να αναφέρεται ως η προτιμώμενη ετικέτα για τη συγκεκριμένη θεματική επικεφαλίδα και οι υπόλοιπες ως εναλλακτικές ετικέτες.

Εφαρμόζοντας το πρότυπο της απλής οργάνωσης της γνώσης Skos οι θεματικές επικεφαλίδες προσδιορίζονται με έναν μόνιμο προσδιοριστή πόρου. Συνδέονται με άλλες θεματικές επικεφαλίδες οι οποίες είναι οργανωμένες σε άτυπες ιεραρχίες, συγκεντρώνονται σε ένα ενιαίο σύστημα και ομαδοποιούνται ως μια συλλογή. Η οργάνωση της συλλογής επιτυγχάνεται με τη χρήση κατάλληλων εργαλείων. Τα εργαλεία αυτά είναι γνωστά ως συστήματα οργάνωσης γνώσης (KOS) όπου χρισιμοποιούνται και ως ελεγχόμενα δομημένα λεξιλόγια.

Η μοντελοποίηση των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου πραγματοποιήθηκε σε δύο αρχεία του excel. Το πρωτεύον αρχείο περιέχει όλες τις θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου στην Ελληνική Γλώσσα στη στήλη Α του excel, οι οποίες είναι ταξινομημένες αλφαβητικά στα Ελληνικά. Ο συνολικός αριθμός των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου στη στήλη Α του excel είναι 9955.

Για κάθε μία θεματική επικεφαλίδα βρέθηκε η αντίστοιχη αγγλική ορολογία και αποθηκεύτηκε σε μια στήλη Β στο πρωτεύον αρχείο του excel. Όπως έχει αναφερθεί παραπάνω κάθε μία θεματική επικεφαλίδα ταυτίζεται με ένα μόνιμο προσδιοριστή πόρου. Μέσω της υπηρεσίας που προσφέρει η βιβλιοθήκη του Κογκρέσου (LCSH), αποθηκεύτηκε ο αντίστοιχος μόνιμος προσδιοριστής πόρου της κάθε θεματικής επικεφαλίδας ως permalink, σε μία στήλη Γ του πρωτεύοντος αρχείου excel. Η εικόνα 1 απεικονίζει μια εκτύπωση οθόνης του πρωτεύοντος αρχείου excel με τις πληροφορίες από τις τρεις στήλες που συλλέχθηκαν μέσω της βιβλιοθήκης του Κογκρέσου.

	A	B	C
1	Θεματική κεφαλίδα στα ελληνικά	Αντίστοιχη αγγλική στη LCSH	Permalink
9703	ΦΥΣΙΟΛΟΓΙΑ	Physiology	http://id.loc.gov/authorities/sh99005103#concept
9704	ΦΥΣΙΟΛΟΓΙΑ (ΑΝΘΡΩΠΙΝΗ)	Human physiology	http://id.loc.gov/authorities/sh85062884#concept
9705	ΦΥΤΑ	Plants	http://id.loc.gov/authorities/sh85102839#concept
9706	ΦΥΤΑ ΕΣΩΤΕΡΙΚΟΥ ΧΩΡΟΥ	House plants	http://id.loc.gov/authorities/sh85062563#concept
9707	ΦΥΤΑ, ΚΑΛΛΙΕΡΓΗΣΙΜΑ	Plants, Cultivated	http://id.loc.gov/authorities/sh85102884#concept
9708	ΦΥΤΕΙΕΣ	Plantations	http://id.loc.gov/authorities/sh85102827#concept
9709	ΦΥΤΙΚΟ ΛΑΔΙ	Vegetable oils	http://id.loc.gov/authorities/sh94004802#concept
9710	ΦΥΤΟΦΑΡΜΑΚΑ	Pesticides	http://id.loc.gov/authorities/sh85100273#concept

Εικόνα 1. Απεικόνιση του πρωτεύοντος αρχείου excel

[4] Πρόσβαση: http://www.xml.com/pub/a/2005/06/22/skos.html Ημ. Πρόσβασης: 20/06/2014

Η παραπάνω εικόνα απεικονίζει στη πρώτη στήλη του πρωτεύοντος αρχείου του excel τις θεματικές επικεφαλίδες στην Ελληνική Γλώσσα, στη στήλη Β την αντίστοιχη Αγγλική ορολογία για κάθε μία από αυτές και στη στήλη Γ τον μόνιμο προσδιοριστή πόρου ως permalink. Με αυτό τον τρόπο καταγράφηκαν συνολικά 871 θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου μέσω της αντίστοιχης υπηρεσίας της βιβλιοθήκης του Κογκρέσου.

Όπως αναφέρθηκε παραπάνω, σε κάθε λεκτικό μιας θεματικής επικεφαλίδας αντιστοιχεί μία ιδιότητα Skos:prefLabel ή Skos:altLabel, η οποία προσδιορίζει αν το εν λόγω λεκτικό είναι η προτιμώμενη ή εναλλακτική απόδοση μιας θεματικής επικεφαλίδας. Για να ορίσουμε τις ιδιότητες αυτές της κάθε μίας θεματικής επικεφαλίδας της βιβλιοθήκης του Ιονίου Πανεπιστημίου, στη φυσική γλώσσα των Ελληνικών, δημιουργήθηκε ένα δεύτερο αρχείο excel. Στο δεύτερο αρχείο excel απομονώθηκαν οι 871 θεματικές επικεφαλίδες και μοντελοποιήθηκαν σε μορφή τριπλού μοτίβου (N-Triples) με τη χρήση της γλώσσας Turtle (TURTLE - Terse RDF Triple Language). Η επόμενη ενότητα περιγράφει τη γλώσσα Turtle και τη μοντελοποίηση των θεματικών επικεφαλίδων του πρωτεύοντος αρχείου με τη χρήση της γλώσσας αυτής.

TURTLE & N-TRIPLE LANGUAGE

Η γλώσσα Turtle (RDF Triple Language) ορίζεται ως μια μορφή έκφρασης των δεδομένων. Ομαδοποιεί τις πληροφορίες χρησιμοποιώντας τριπλέτες (triples). Σε κάθε μία από τις τρεις θέσεις της τριπλέτας βρίσκεται ένα υποκείμενο, ένα κατηγόρημα και ένα αντικείμενο αντίστοιχα. Καθένα από αυτά τα στοιχεία ταυτίζεται με ένα μόνιμο προσδιοριστή πόρου (Turtle W3C WG, 2014).

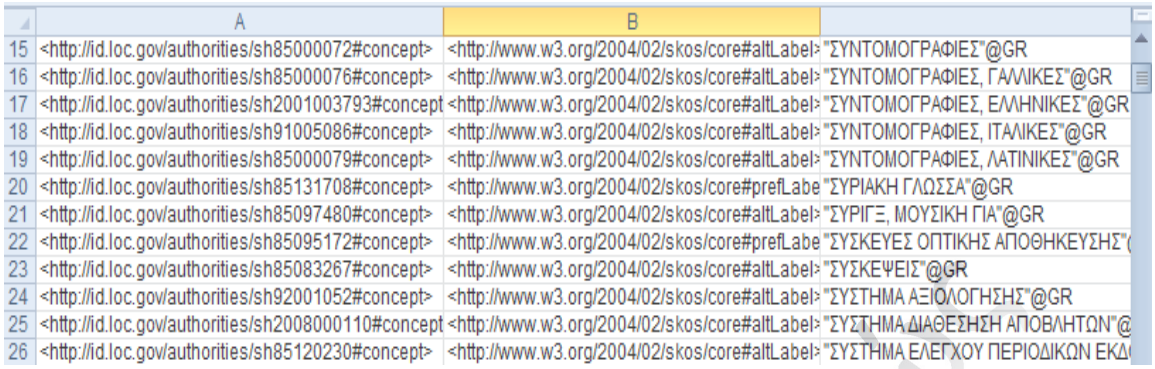
Ένα έγγραφο Turtle επιτρέπει να γραφτεί μια συμπαγή μορφή κειμένου. Η μορφή κειμένου αποτελείται από μια σειρά σχολίων ή τριπλέτες ή κενές γραμμές. Τα σχόλια μπορεί να δοθούν μετά τη δίεση '#' και να συνεχίζονται μέχρι το τέλος της γραμμής αυτής.

Κάθε θεματική επικεφαλίδα μοντελοποιήθηκε σε μία γραμμή σε μορφή τριπλέτας, που τερματίζεται με την τελεία '.'. Αυτό αντιστοιχεί σε N-Τριπλέτες (N-Triples).

Ένα ενιαίο αναγνωριστικό πόρου περικλείεται σε μια ετικέτα, αρχίζοντας με το δεξί εισαγωγικό '<' και κλείνοντας με το αριστερό εισαγωγικό '>'.

Τα λεκτικά μπορούν να αποδοθούν με ένα επίθεμα γλώσσας ή με ένα τύπο δεδομένων ενός ενιαίου αναγνωριστικού πόρου, αλλά όχι και τα δύο. Στην εργασία μας χρησιμοποιήσαμε το επίθεμα της φυσικής γλώσσας 'Ελληνικά'. Η φυσική γλώσσα βρίσκεται μέσα σε διπλά εισαγωγικά '" "' και στη συνέχεια ακολουθεί το σύμβολο @ με την ετικέτα της φυσικής γλώσσας 'GR'.

Εφαρμόζοντας τους παραπάνω απλούς κανόνες της γλώσσας Turtle στις θεματικές επικεφαλίδες του πρωτεύοντος αρχείου excel, δημιουργήθηκε ένα δεύτερο αρχείο του excel με τις απομονωμένες 871 θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου. Το νέο αρχείο του excel μοντελοποιήθηκε περιέχοντας στη στήλη Α τον αντίστοιχο μόνιμο προσδιοριστή πόρου της κάθε θεματικής επικεφαλίδας. Όλες οι θεματικές επικεφαλίδες ανήκουν στην κλάση Skos:Concept. Η στήλη Β περιέχει την επιλογή προτιμώμενης ή εναλλακτικής απόδοσης της ετικέτας κάθε μίας θεματικής επικεφαλίδας (altlabel, preflabel). Οι πληροφορίες που βρίσκονται στις στήλες Α και Β συλλέχθηκαν μέσω της αντίστοιχης υπηρεσίας που προσφέρει η βιβλιοθήκη του Κογκρέσου. Η στήλη Γ περιέχει το λεκτικό της κάθε μίας θεματικής επικεφαλίδας στα Ελληνικά. Η παρακάτω εικόνα απεικονίζει μια εκτύπωση οθόνης προερχόμενη από το δευτερεύον αρχείο του excel με τις απομονωμένες θεματικές επικεφαλίδες του πρωτεύοντος αρχείου excel.



	A	B	
15	<http://id.loc.gov/authorities/sh85000072#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ"@GR
16	<http://id.loc.gov/authorities/sh85000076#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ, ΓΑΛΛΙΚΕΣ"@GR
17	<http://id.loc.gov/authorities/sh2001003793#concept	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ, ΕΛΛΗΝΙΚΕΣ"@GR
18	<http://id.loc.gov/authorities/sh91005086#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ, ΙΤΑΛΙΚΕΣ"@GR
19	<http://id.loc.gov/authorities/sh85000079#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ, ΛΑΤΙΝΙΚΕΣ"@GR
20	<http://id.loc.gov/authorities/sh85131708#concept>	<http://www.w3.org/2004/02/skos/core#prefLabe	"ΣΥΡΙΑΚΗ ΓΛΩΣΣΑ"@GR
21	<http://id.loc.gov/authorities/sh85097480#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΡΙΓΞ, ΜΟΥΣΙΚΗ ΓΙΑ"@GR
22	<http://id.loc.gov/authorities/sh85095172#concept>	<http://www.w3.org/2004/02/skos/core#prefLabe	"ΣΥΣΚΕΥΕΣ ΟΠΤΙΚΗΣ ΑΠΟΘΗΚΕΥΣΗΣ"(
23	<http://id.loc.gov/authorities/sh85083267#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΣΚΕΨΕΙΣ"@GR
24	<http://id.loc.gov/authorities/sh92001052#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΣΤΗΜΑ ΑΞΙΟΛΟΓΗΣΗΣ"@GR
25	<http://id.loc.gov/authorities/sh2008000110#concept	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΣΤΗΜΑ ΔΙΑΘΕΣΗΣΗ ΑΠΟΒΛΗΤΩΝ"@
26	<http://id.loc.gov/authorities/sh85120230#concept>	<http://www.w3.org/2004/02/skos/core#altLabel>	"ΣΥΣΤΗΜΑ ΕΛΕΓΧΟΥ ΠΕΡΙΟΔΙΚΩΝ ΕΚΔ(

Εικόνα 2. Εκτύπωση του δευτερεύον αρχείου excel με τις Θεματικές επικεφαλίδες σε μορφή N-Triples

Στη στήλη Δ του δευτερεύοντος αρχείου του excel έχει προστεθεί η τελεία '.', η οποία δεν είναι ευδιάκριτη στην παραπάνω εικόνα. Συνοψίζοντας, η κάθε γραμμή του δευτερεύοντος αρχείου του excel έχει την παρακάτω μορφή η οποία δηλώνει το θέμα – κατηγόρημα – αντικείμενο (subject - predicate - object):

<http://id.loc.gov/authorities/sh85136516#concept> <http://www.w3.org/2004/02/skos/core#altLabel> "ΣΥΝΤΕΧΝΙΕΣ"@GR .

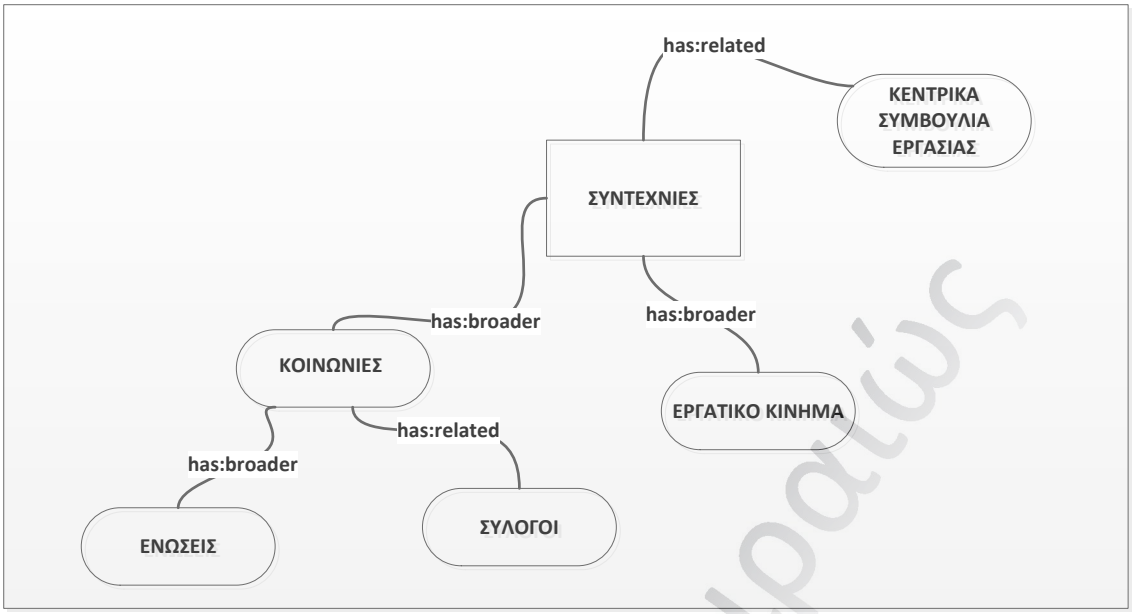
Το πρόβλημα που καλούμαστε να αντιμετωπίσουμε στην παρούσα μεταπτυχιακή διατριβή είναι να βρεθεί η συνδετική δομή μεταξύ των 871 θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου. Πιο συγκεκριμένα θέλουμε να βρεθεί για κάθε μία θεματική επικεφαλίδα ο ευρύτερος, στενότερος και συναφής θεματικός όρος αυτής.

Προκειμένου να επιτευχθεί αυτό χρησιμοποιήθηκε η αντίστοιχη υπηρεσία της βιβλιοθήκης του Κογκρέσου ως ένα λεξιλόγιο θεματικών όρων αντλώντας σε πραγματικό χρόνο τις πληροφορίες που δεν έχουμε στη δική μας βάση δεδομένων. Στο επόμενο κεφάλαιο περιγράφεται πως κατασκευάστηκε η υπηρεσία ανάκτησης συνδεδεμένης δομής καταλόγου, η οποία εμφανίζει τις σχέσεις που προκύπτουν μεταξύ των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου μέσω της βιβλιοθήκης του Κογκρέσου.

ΥΠΗΡΕΣΙΑ ΑΥΤΟΜΑΤΗΣ ΑΝΑΚΤΗΣΗΣ ΣΥΝΔΕΔΕΜΕΝΗΣ ΔΟΜΗΣ ΚΑΤΑΛΟΓΟΥ

Η υπηρεσία αυτόματης ανάκτησης συνδεδεμένης δομής καταλόγου βασίζεται σε τεχνολογίες του σημασιολογικού ιστού. Μέσω της βιβλιοθήκης του Κογκρέσου ο χρήστης ανακαλύπτει τη συνδετική δομή μεταξύ των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου. Πιο συγκεκριμένα για κάθε μία θεματική επικεφαλίδα προσδιορίζονται οι ευρύτεροι, στενότεροι και συναφείς θεματικοί όροι (BT, NT, RT). Η κάθε μία θεματική επικεφαλίδα μπορεί να έχει έναν ή περισσότερους ευρύτερους, στενότερους ή συναφείς θεματικούς όρους ή ακόμα και κανέναν. Οι ευρύτεροι, στενότεροι και συναφείς θεματικοί όροι παραπέμπουν σε άλλες θεματικές επικεφαλίδες οι οποίες αντιστοιχούν με την σειρά τους σε ένα ενιαίο αναγνωριστικό πόρου.

Η παρακάτω απεικονίζει σε ένα γράφο τις σχέσεις που προκύπτουν από τη θεματική επικεφαλίδα "ΣΥΝΤΕΧΝΙΕΣ"@GR.



Εικόνα 3. Γράφος 1

Όπως προκύπτει από την παραπάνω εικόνα η θεματική επικεφαλίδα έχει ως ευρύτερους θεματικούς όρους τις θεματικές επικεφαλίδες "ΚΟΙΝΩΝΙΕΣ" και "ΕΡΓΑΤΙΚΟ ΚΙΝΗΜΑ". Ως συναφή θεματικό όρο τα "ΚΕΝΤΡΙΚΑ ΣΥΜΒΟΥΛΙΑ ΕΡΓΑΣΙΑΣ". Αντίστοιχα η θεματική επικεφαλίδα "ΚΟΙΝΩΝΙΕΣ" έχει ως ευρύτερο θεματικό όρο τις "ΕΝΩΣΕΙΣ" και ως συναφή θεματικό όρο τους "ΣΥΛΟΓΟΥΣ".

Η υπηρεσία ανάκτησης συνδεδεμένης δομής χρησιμοποιεί τις τεχνολογίες σημασιολογικού ιστού προκειμένου να βρεθούν οι συνδεδεμένες σχέσεις όπως απεικονίζεται στο παραπάνω γράφο. Καταγράφει τη συνδετική δομή των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου μέσω της αντίστοιχης υπηρεσίας τη Βιβλιοθήκης του Κογκρέσου.

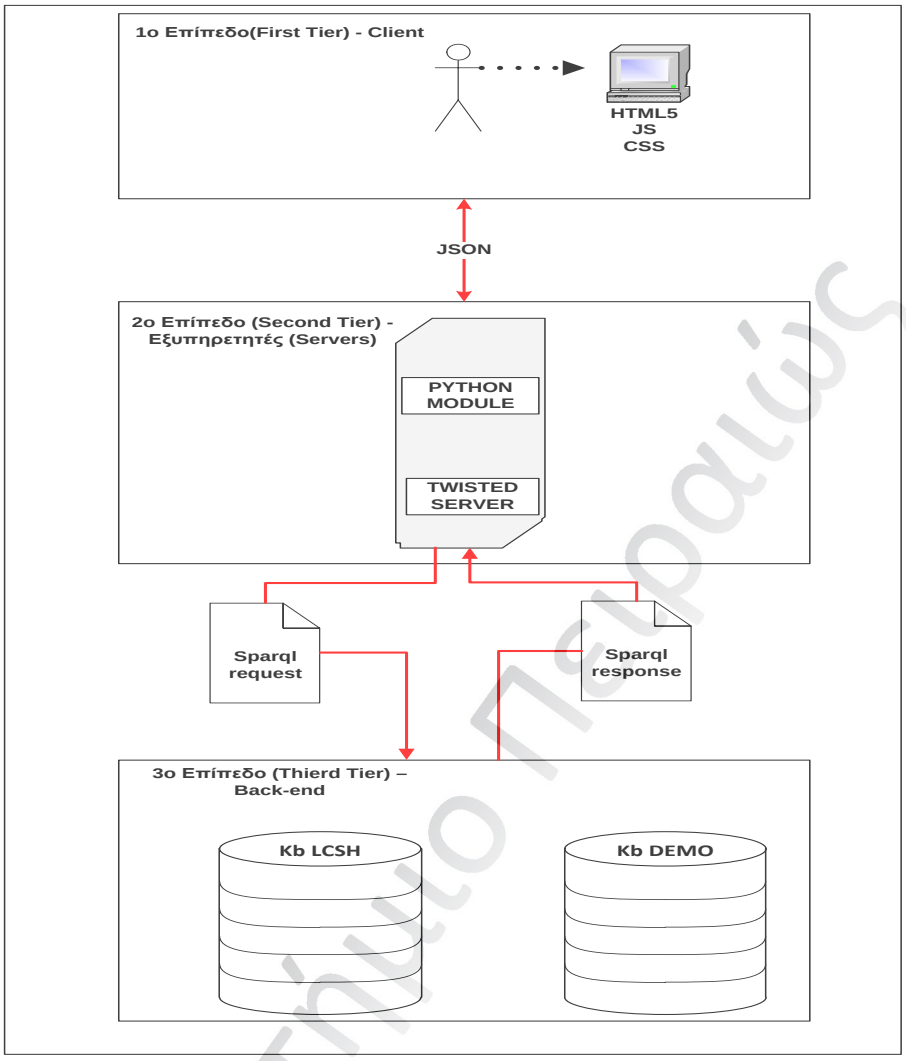
Αρχικά μοντελοποιήθηκαν οι θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου σύμφωνα με το πρότυπο του απλού συστήματος οργάνωσης θεματικών επικεφαλίδων Skos, όπως περιγράψαμε στο προηγούμενο κεφάλαιο. Έπειτα από τη μοντελοποίηση των θεματικών επικεφαλίδων δημιουργήθηκε μια εξειδικευμένη βάση δεδομένων με τις πλέον μοντελοποιημένες θεματικές επικεφαλίδες σε μορφή τριπλετών (triple pattern). Χρησιμοποιήθηκε η γλώσσα προγραμματισμού python προκειμένου να διαχειριστεί τα sparql ερωτήματα έτσι ώστε να ανακαλυφθεί η συνδετική δομή των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου από την αντίστοιχη υπηρεσία της βιβλιοθήκης του Κογκρέσου. Η υπηρεσία βασίζεται στην αρχιτεκτονική τριών επιπέδων 3-Tier η οποία περιγράφεται στο παρακάτω κεφάλαιο.

ΑΡΧΙΤΕΚΤΟΝΙΚΗ 3-TIER

Η υπηρεσία αυτόματης ανάκτησης συνδεδεμένης δομής θεματικών επικεφαλίδων βασίζεται στην αρχιτεκτονική 3-tier. Η αρχιτεκτονική 3-tier αποτελείται από τρία επίπεδα[5]:

- 1ο Επίπεδο (First Tier) - Εμφάνιση στον πελάτη (Client)
- 2ο Επίπεδο (Second Tier) – Εξυπηρετητής (Server)
- 3ο Επίπεδο (Third Tier) – Αποθήκευση Δεδομένων (Back-end)

[5] Πρόσβαση: http://www.altec.gr/index.php/technology/3-tier-client-server.html Ημ. πρόσβασης: 20/06/2014



Εικόνα 4. Αρχιτεκτονική τριών επιπέδων – 3 –Tier

Η παραπάνω εικόνα απεικονίζει τα τρία επίπεδα της υπηρεσίας αυτόματης ανάκτησης συνδεδεμένης δομής καταλόγου. Το πρώτο επίπεδο αποτελεί τη επαφή του χρήστη με το σύστημα, το δεύτερο επίπεδο αποτελεί το κύριο τμήμα του λογισμικού, ενώ το τρίτο επίπεδο αποτελεί τον αποθηκευτικό χώρο του συστήματος.

Η συγκρότηση του συστήματος σε τρία επίπεδα εξασφαλίζει την ελαχιστοποίηση της επιβάρυνσης του δικτύου, λόγω μεταφοράς μεγάλου όγκου δεδομένων, όπως για παράδειγμα την εκτέλεση ενός Sparql ερωτήματος για την ανάκτηση των πληροφοριών από τη βάση δεδομένων της βιβλιοθήκης του Κογκρέσου στην οποία είναι αποθηκευμένες χιλιάδες εγγραφές, εμφανίζοντας στο χρήστη μόνο το αποτέλεσμα.

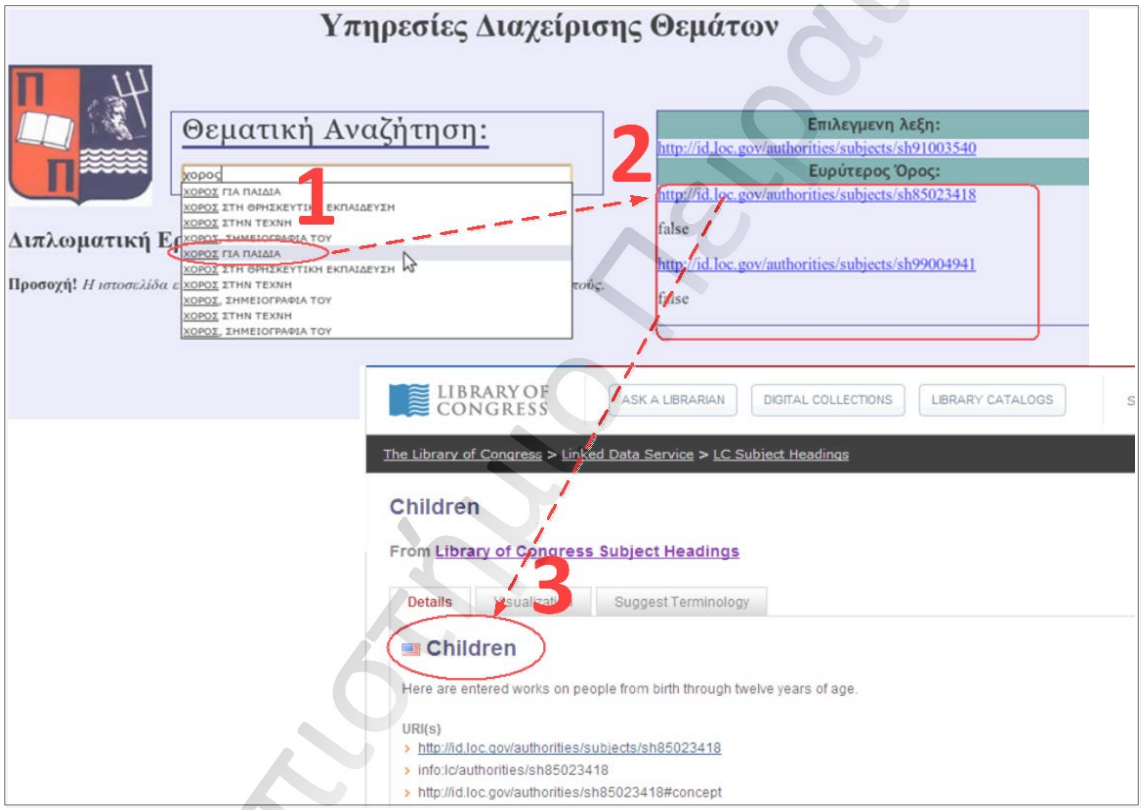
Επιπροσθέτως παρέχεται η δυνατότητα διαχωρισμού του διακομιστή δεδομένων (Database Server), από τούς διακομιστές της εφαρμογής (Application Servers), ώστε να εκτελούνται σε διαφορετικές διεργασίες. Κατά συνέπεια, ο καθορισμός των κρίσιμων μεγεθών απόδοσης των αντίστοιχων διεργασιών μπορεί να γίνεται ανεξάρτητα, ενώ παράλληλα εξασφαλίζεται απεριόριστη επεκτασιμότητα, χωρίς ανακατασκευή, του λογισμικού.

Τέλος, επιτρέπεται η επικοινωνία μεταξύ των διακομιστών δια μέσου πρωτοκόλλου HTTP. Στην επόμενη ενότητα περιγράφεται η λειτουργία της υπηρεσίας για καθέ επιπέδο χωριστά[5].

ΠΡΩΤΟ ΕΠΙΠΕΔΟ – **ΕΜΦΑΝΙΣΗ ΣΤΟΝ ΠΕΛΑΤΗ** (CLIENT)

Το πρώτο επίπεδο αποτελεί την επαφή του χρήστη με την υπηρεσία (User Interface). Στο επίπεδο αυτό, πραγματοποιείται η διαχείριση των οθονών εργασίας, καθώς επίσης και η μορφοποίηση των δεδομένων που εμφανίζονται. Η επικοινωνία του πελάτη (Client) με την εφαρμογή (Application) του δεύτερου επιπέδου, πραγματοποιείται κάνοντας χρήση ενός μόνο πακέτου δεδομένων κάθε φορά[5].

Μόλις ο χρήστης πληκτρολογήσει τα αρχικά μιας λέξης σε ένα πλαίσιο αναζήτησης κειμένου, η υπηρεσία του εμφανίζει αυτόματα μία λίστα με προτεινόμενες λέξεις. Ο χρήστης επιλέγει μία θεματική επικεφαλίδα από την προτεινόμενη λίστα. Επιλέγοντας μια θεματική επικεφαλίδα, αν προκύψουν αποτελέσματα, η υπηρεσία τα εμφανίζει σε ένα πίνακα αποτελεσμάτων στο δεξί μέρος της οθόνης. Τα αποτελέσματα που παραπέμπουν σε άλλες θεματικές επικεφαλίδες, απεικονίζονται με τη μορφή υπέρ-συνδέσμου. Η παρακάτω εικόνα απεικονίζει την παραπάνω λειτουργικότητα μέσα από ένα παράδειγμα:



Εικόνα 5. Λειτουργία της Υπηρεσίας Διαχείρισης Θεματικών επικεφαλίδων

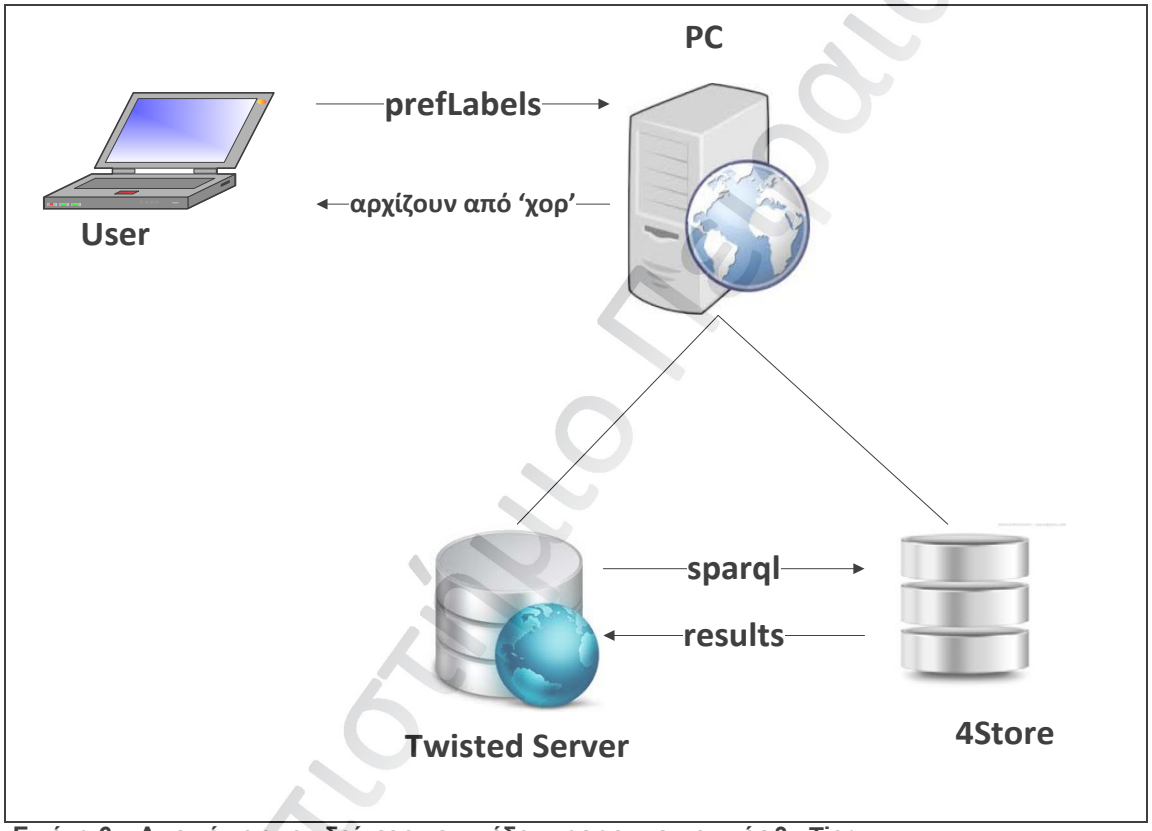
Στο παράδειγμα της παραπάνω εικόνας ο χρήστης μόλις αρχίσει να πληκτρολογεί τη λέξη 'χορός', η υπηρεσία αυτόματα εμφανίζει μία λίστα με προτεινόμενες λέξεις (No 1). Επιλέγοντας το θεματικό όρο 'χορός για παιδιά' η υπηρεσία εμφανίζει στο δεξί μέρος της οθόνης τα αποτελέσματα μέσα σε ένα πίνακα (No 2). Παρατηρούμε ότι έπειτα από κάθε αποτέλεσμα ακολουθεί η λέξη 'false', αυτό σημαίνει πως το αποτελέσμα δεν υπάρχει στη δική μας εξειδικευμένη βάση δεδομένων (Kb Demo). Τέλος επιλέγοντας ένα οποιοδήποτε αποτέλεσμα, η υπηρεσία παραπέμπει σε μία νέα καρτέλα στην αντίστοιχη σελίδα της βιβλιοθήκης του Κογκρέσου. Στο παράδειγμα της εικόνας που παρατέθηκε επιλέγοντας το πρώτο αποτέλεσμα του ευρύτερου θεματικού όρου, αντιστοιχεί στο λεκτικό 'Children' (No 3).

Για τις εικόνες και τη δομή της υπηρεσίας χρησιμοποιήθηκε η υπέρ γλώσσα σήμανσης κειμένου (HTML5 – Hypertext Markup Language). Για να τον ορισμό της εμφάνισης του περιεχομένου χρησιμοποιήθηκαν επικαλυπτόμενα φύλλα στυλ (CSS3- Cascading Style Sheets). Για την αλληλεπίδραση του χρήστη με την υπηρεσία έχει ενσωματώσει στην υπέρ γλώσσα σήμανσης κειμένου η δυναμική γλώσσα προγραμματισμού javascript. Τέλος, για τη μετάδοση των δεδομένων μεταξύ του εξυπηρετητή με την υπηρεσία ανάκτησης συνδετικής

δομής καταλόγου, του πρώτου και του δεύτερου επιπέδου της αρχιτεκτονικής 3-Tier, χρησιμοποιήθηκε η μορφή δεδομένων json (JavaScript Object Notation).

ΔΕΥΤΕΡΟ ΕΠΙΠΕΔΟ – ΕΞΥΠΗΡΕΤΗΤΗΣ (SERVER)

Το δεύτερο επίπεδο εξυπηρετεί τα αιτήματα των χρηστών όπως αυτά διατυπώνονται από το πρώτο επίπεδο. Συντονίζει την υπηρεσία με τη χρήση του εξυπηρετητή twisted και επεξεργάζεται εντολές με τη χρήση της γλώσσας προγραμματισμού python. Επιπλέον υπάρχει μία επικοινωνία μεταξύ πρώτου και δεύτερου επιπέδου, ανάμεσα στην υπηρεσία και στον εξυπηρετητή, με τη μορφή δεδομένων json (JavaScript Object Notation). Η παρακάτω εικόνα απεικονίζει τη ροή πληροφοριών του δεύτερου επιπέδου.



Εικόνα 6 – Απεικόνιση του δεύτερου επιπέδου της αρχιτεκτονικής 3 - Tier

Μόλις ο χρήστης αρχίσει να πληκτρολογεί έναν θεματικό όρο, όπως περιγράψαμε στο παράδειγμα της εικόνας 5 της προηγούμενης ενότητας, ο εξυπηρετητής twisted που είναι εγκατεστημένος τοπικά στον υπολογιστή μας, επικοινωνεί με το 4Store. Ο εξυπηρετητής twisted στέλνει στο 4Store ένα Sparql ερώτημα. Ζητάει από το 4Store να του απαντήσει αναζητώντας όλους τους προτιμώμενους θεματικούς όρους (preferable labels). Με αυτόν τον τρόπο επιστρέφεται στην οθόνη εργασίας και εμφανίζεται στο χρήστη ό,τι ξεκινά από ΄χορ΄.

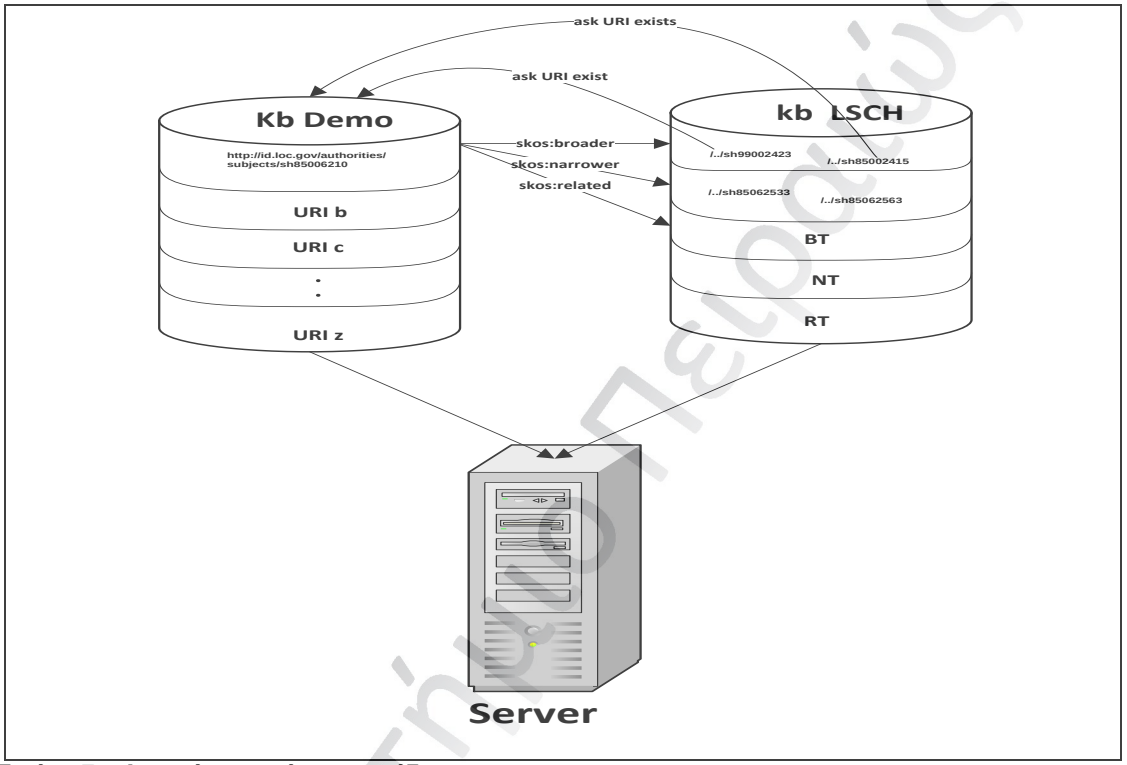
Τα αποτελέσματα επιστρέφονται στον εξυπηρετητή twisted σε μορφή επεκτάσιμης γλώσσας σήμανσης κειμένου (XML – Extensible Markup Language). Ο εξυπηρετητής διαβάζει τα αποτελέσματα της επεκτάσιμης γλώσσας σήμανσης κειμένου μετατρέποντάς τα σε ευανάγνωστη μορφή προς το χρήστη.

ΤΡΙΤΟ ΕΠΙΠΕΔΟ – ΑΠΟΘΗΚΕΥΣΗ ΔΕΔΟΜΕΝΩΝ (BACK-END)

Το εργαλείο ανοικτού κώδικα 4Store, παρέχει όλες τις απαραίτητες λειτουργίες για την αποθήκευση, ανάκτηση, ενημέρωση και συντήρηση των δεδομένων του συστήματος καθώς

επίσης και όλους τους απαραίτητους μηχανισμούς για την ακεραιότητα των δεδομένων (Data Integrity)[5].

Η ανάκτηση των δεδομένων πραγματοποιείται μέσω επερωτήσεων της γλώσσας Sparql, ερωτήματα που είναι ενσωματωμένα στο κυρίως πρόγραμμα της εφαρμογής, υλοποιημένα στη γλώσσα προγραμματισμού python. Επιπλέον με τη χρήση της γλώσσας python επιτυγχάνεται η επικοινωνία μεταξύ της εξειδικευμένης βάσης δεδομένων 4Store με την αντίστοιχη βάση που διαθέτει η βιβλιοθήκη του Κογκρέσου. Η παρακάτω εικόνα απεικονίζει την ανταλλαγή των πληροφοριών μεταξύ των δύο βάσεων δεδομένων:



Εικόνα 7 – Απεικόνιση τρίτου επιπέδου

Η παραπάνω εικόνα απεικονίζει την επικοινωνία μεταξύ της εξειδικευμένης βάσης δεδομένων 4Store, στην οποία είναι αποθηκευμένες οι θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου, έπειτα από τη μοντελοποίηση του προτύπου της απλής οργάνωσης της γνώσης Skos, με την αντίστοιχη βάση δεδομένων της υπηρεσίας που παρέχει η βιβλιοθήκη του Κογκρέσου.

Όπως βλεπουμε, στο αριστερό μέρος της παραπάνω εικόνας βρίσκεται η εξειδικευμένη βάση δεδομένων (Kb Demo). Οι πληροφορίες που έχουμε για τις θεματικές επικεφαλίδες σε αυτή τη βάση αποθήκευσης δεδομένων είναι τα ενιαία αναγνωριστικά πόρου, η ιδιότητα τους (προτιμώμενες ή εναλλακτικές), και οι αντίστοιχοι θεματικοί όροι ως λεκτικά επιθέματα της Ελληνικής γλώσσας για για τις 871 θεματικές επικεφαλίδες της βιβλιοθήκης του Ιονίου Πανεπιστημίου. Στο δεξί μέρος της εικόνας 7 απεικονίζεται η βάση της βιβλιοθήκης του Κογκρέσου (kb LCSH), όπου βρίσκεται η επιπλέον πληροφορία των ευρύτερων στενότερων και συναφών θεματικών όρων για 4.029.085 θεματικές επικεφαλίδες.

Για κάθε θεματική επικεφαλίδα η εξειδικευμένη βάση δεδομένων (kb Demo) ζητάει από την αντίστοιχη βάση της βιβλιοθήκης του Κογκρέσου τους ευρύτερους, στενότερους και συναφείς θεματικούς όρους. Για κάθε ένα αποτέλεσμα που προκύπτει, γίνεται αναζήτηση στην εξειδικευμένη βάση δεδομένων (Kb Demo).

Συνεπώς με αυτό τον τρόπο βρίσκεται η συνδετική δομή των θεματικών επικεφαλίδων της βιβλιοθήκης του Ιονίου Πανεπιστημίου. Στην επόμενη ενότητα γίνεται ανάλυση της γλώσσας επερωτήσεων Sparql.

SPARQL **ΓΛΩΣΣΑ ΕΠΕΡΩΤΗΣΕΩΝ ΓΙΑ ΤΑ** RDF **ΔΕΔΟΜΕΝΑ**

Η Sparql είναι γλώσσα επερωτήσεων και χρησιμοποιείται για την δημιουργία Sparql ερωτημάτων για τα δεδομένα που είναι σε μορφή πλαισίου περιγραφής πόρου (RDF). Είναι δηλαδή μια γλώσσα επερωτήσεων για βάσεις δεδομένων, που μοιάζει με την γνωστή γλώσσα SQL (SQL – Structured Query Language). Είναι σε θέση να χειριστεί και να ανακτήσει δεδομένα που είναι αποθηκευμένα σε μορφή RDF. Από το 2008 έως και σήμερα αναγνωρίζεται ως μια από τις βασικές λειτουργίες του σημασιολογικού ιστού (DuCharm, 2013).

Το μοτίβο που ακολουθούν τα Sparql ερωτήματα αποτελείται από το κυρίως σώμα και από την κεφαλή η οποία εκφράζει το πώς να κατασκευαστεί η απάντηση σε ένα ερώτημα.

Στη περιοχή της δήλωσης του χώρου των ονομάτων χρησιμοποιούνται συντομεύσεις όπως απεικονίζονται στον παρακάτω πίνακα:

Συντομογραφίες	URI
Skos:	**http://www.w3.org/2004/02/Skos/core#**
rdf:	**http://www.w3.org/1999/02/22-rdf-syntax-ns#**

Πίνακας 1 – Uri Συντομογραφίες (abbreviations)

Σύμφωνα με τον παραπάνω πίνακα, στο τμήμα των δεδομένων δηλώνονται τα αναγνωριστικά πόρου του προτύπου rdfs και Skos τα οποία περικλείονται μέσα σε άγκιστρα. Ακολουθεί στη κεφαλή του ερωτήματος το τμήμα του αποτελέσματος. Το κυρίως σώμα του ερωτήματος είναι το τριπλό μοτίβο με μεταβλητές, οι οποίες δηλώνονται με το σύμβολο του ερωτηματικού ‘ ? ’ (Arenas & Perez, 2011).

Στην εξειδικευμένη τοπική βάση δεδομένων όπως έχουμε αναφέρει στα προηγούμενα κεφάλαια το τριπλό μοτίβο αποτελείται από το ενιαίο αναγνωριστικό πόρου, την ιδιότητα της θεματικής επικεφαλίδας και την ονομασία της κάθε θεματικής επικεφαλίδας στην Ελληνική γλώσσα ως λεκτικό επίθεμα.

Παραδείγματος χάριν το τριπλό μοτίβο της θεματικής επικεφαλίδας ‘ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ’ που βρίσκεται στην εξειδικευμένη τοπική βάση είναι το παρακάτω:

```
<http://id.loc.gov/authorities/subjects/sh85006210.html#Concept> <http://www.w3.org/2004/02/skos/core#altLabel> "ΥΔΑΤΟΚΑΛΛΙΕΡΓΙΑ"@GR
```

Για να δούμε αν η θεματική επικεφαλίδα ‘ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ’ υπάρχει στην εξειδικευμένη τοπική βάση δεδομένων (Kb Demo) διατυπώνουμε το παρακάτω ερώτημα:

Data Segment

declare prefix shortcuts (optional)

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
```

Query result segment

```
ASK where
{
```

define the dataset of tripple pattern

```
<http://id.loc.gov/authorities/subjects/sh85006210> ?p
?o .}
```

Στη δήλωση χώρου των ονομάτων δηλώνεται το πρόθεμα rdfs και Skos όπως προκύπτει από τον πίνακα 1 των συντομογραφιών της παραπάνω σελίδας. Το κυρίως μέρος

του ερωτήματος είναι το σύνολο του τριπλού μοτίβου με τις μεταβλητές με το σύμβολο του ερωτηματικού '?' καθώς προηγείται το τμήμα αποτελέσματος του ερωτήματος.

Για να εντοπιστούν ποιοι είναι οι ευρύτεροι θεματικοί όροι της θεματικής επικεφαλίδας 'ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ' από τη βάση της βιβλιοθήκης του Κογκρέσου (Kb LCSH), το Sparql ερώτημα που προκύπτει είναι το παρακάτω:

Data Segment

PREFIX Skos:<http://www.w3.org/2004/02/Skos/core#>

Query Segment

SELECT ?o where {

<http://id.loc.gov/authorities/subjects/sh85006210> Skos:broader?o.}

Για την εκτέλεση των παραπάνω ερωτημάτων χρησιμοποιήθηκε το 4Store. Στην παρακάτω ενότητα γίνεται ανάλυση της χρήσης του εργαλείου 4Store για τα RDF δεδομένα.

4STORE **ΓΙΑ ΤΑ** RDF **ΔΕΔΟΜΕΝΑ**

Το εργαλείο 4Store σχεδιάστηκε από τον Steve Harris της εταιρίας Garlik για την υποστήριξη εφαρμογών σημασιολογικού ιστού (Semantic Web Applications). Η πλατφόρμα ανοιχτού κώδικα παρέχεται δωρεάν περίπου εδώ και τρία χρόνια. Είναι ένα εργαλείο αποθήκευσης βάσεων δεδομένων (Sparql HTTP protocol server), αλλά παράλληλα προσφέρει μια μηχανή διαχείρισης Sparql ερωτημάτων (Sparql Queries) για τα RDF δεδομένα. Τα κύρια χαρακτηριστικά της μηχανής αυτής είναι η απόδοση, η επεκτασιμότητα και η σταθερότητα. Είναι μηχανή αποθήκευσης μόνο για RDF δεδομένα και τίποτε παραπάνω. Δεν έχει την δυνατότητα αποθήκευσης Sparql ερωτημάτων, παρά μόνο την εκτέλεσή τους.

Το 4Store έχει αποδειχθεί ότι είναι ασφαλές, με λίγα χαρακτηριστικά αλλά με καλές επιδόσεις και εύκολο στη χρήση. Η πλατφόρμα είναι γραμμένη σε ANSI C99 και σχεδιασμένη να τρέχει κυρίως σε λειτουργικό Unix και πλατφόρμες Linux. Η εγκατάσταση είναι εύκολη ενώ για την έκδοση του λειτουργικού συστήματος Ubuntu χρειάστηκε να γίνει εγκατάσταση των βιβλιοθηκών Raptor και Rasqal[6].

Από το δευτερεύον αρχείο του excel με τις θεματικές επικεφαλίδες της Βιβλιοθήκης του Ιονίου Πανεπιστημίου σε μορφή N-Triples, δημιουργήθηκε ένα αρχείο με επέκταση .n3 και φορτώθηκε στο 4Store με τη χρήση εντολών από το τερματικό του υπολογιστή μας. Ο Sparql HTTP protocol server τρέχει τοπικά στον υπολογιστή μας στην TCP port 8000.

Για να ξεκινήσει ο εξυπηρετητής πληκτρολογούμε στο τερματικό την παρακάτω εντολή *4s-httpd –p 8000*. Πληκτρολογώντας την εντολή το 4s-httpd γίνεται σύνδεση το 4Store μέσω του τελικού σημείου (back-end) του Sparql εξυπηρετητή.

Η εξειδικευμένη βάση που δημιουργήσαμε (Kb Demo), περιέχει 871 επεξεργασμένες θεματικές επικεφαλίδες σε μορφή N-Triples. Για την εκτέλεση ενός Sparql ερωτήματος ανοίγουμε ένα πρόγραμμα περιήγησης (Browser) και πληκτρολογούμε την παρακάτω διεύθυνση *http://localhost:8000/test*.

Στην εξειδικευμένη βάση (Kb Demo) που βρίσκεται τοπικά στον υπολογιστή μας, οι μόνες πληροφορίες που έχουμε για την κάθε θεματική επικεφαλίδα είναι ο μόνιμος προσδιοριστής και οι ιδιότητές τους (alternative ή preferable Label).

[6] Πρόσβαση: http://4store.org/
Ημ. Πρόσβασης: 20/06/2013

Με τη χρήση του 4Store μπορούμε να συνδεθούμε στη βάση της βιβλιοθήκης του Κογκρέσου (Kb LCSH), πληκτρολογώντας σε μια δεύτερη καρτέλα του προγράμματος περιήγησης (New Tab) τη διεύθυνση http://83.212.99.113:8089/test/ η οποία περιέχει 4.029.085 θεματικές επικεφαλίδες.

Τα αποτελέσματα από τα Sparql ερωτήματα εξάγονται από το 4Store σε μορφή XML:

```
ASK WHERE {
<http://id.loc.gov/authorities/subjects/sh85006210> ?p ?o.}
```

Το αποτέλεσμα του παραπάνω ερωτήματος που παράγει το εργαλείο διαχείρισης sparql ερωτημάτων είναι σε μορφή XML:

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
 <head>
 </head>
 <boolean>true</boolean>
</sparql>
```

Δεδομένου ότι η απάντηση του παραπάνω ερωτήματος είναι 'true', η θεματική επικεφαλίδα 'ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ' υπάρχει στην εξειδικευμένη τοπική βάση δεδομένων.

Το δεύτερο παράδειγμα του Sparql ερωτήματος της προηγούμενης ενότητας ήταν να βρούμε όλους τους ευρύτερους θεματικούς όρους της επικεφαλίδας 'ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ' από την αντίστοιχη υπηρεσία της βιβλιοθήκης του Κογκρέσου (Kb LCSH):

```
SELECT ?o where {
<http://id.loc.gov/authorities/subjects/sh85006210> Skos:broader ?o.}
```

Έπειτα από την εκτέλεση του παραπάνω sparql ερωτήματος στην αντίστοιχη υπηρεσία της βιβλιοθήκης του Κογκρέσου το αποτέλεσμα είναι το παρακάτω:

```
<?xml version="1.0"?>
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
   <variable name="o"/>
  </head>
  <results>
   <result>
<binding name="o"><uri>http://id.loc.gov/authorities/subjects/sh99002423</uri></binding>
   </result>
```

```
    <result>
      <binding
name="o"><uri>http://id.loc.gov/authorities/subjects/sh85002415</uri></binding>
    </result>
  </results>
</sparql>
```

Η θεματική επικεφαλίδα 'ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ' έχει δύο ευρύτερους όρους. Προκειμένου να μετατραπούν τα αποτελέσματα που παράγει το 4Store σε μορφή ευανάγνωστη προς τους χρήστες κατασκευάστηκε ένας απλός αλγόριθμος με την χρήση της γλώσσας python όπου αναλύεται στην επόμενη ενότητα.

ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ PYTHON

Η python είναι γλώσσα υψηλού επιπέδου, απλή στην εκμάθησή της για τους αρχάριους και με εξαιρετικές ικανότητες για τους προχωρημένους. Αντικειμενοστραφής, διερμηνευτική, γλώσσα ανοιχτού κώδικα, προ-εγκατεστημένη στο λειτουργικό σύστημα Linux.

Διαχειρίζεται σύνθετα αντικείμενα τα οποία χρησιμοποιούμε για όλες τι δομές που θέλουμε να υλοποιήσουμε. Τα δύο πιο βασικά είναι οι λίστες (Lists) και τα λεξικά (Dictionaries). Οι δύο αυτοί τύποι των αντικειμένων μπορούν να αλλάξουν περιεχόμενα (Mutable)[7].

Με τη χρήση της βιβλιοθήκης *4Store httpd* η python μπορεί να διαχειριστεί, να διαβάσει και να απεικονίσει τα sparql αποτελέσματα σε μορφή XML για το 4Store. Με τη χρήση της βιβλιοθήκης *urllib2* επιστρέφεται το περιεχόμενο μιας ιστοσελίδας. Με τη διάταξη δεδομένων json, η python υποδέχεται την απαιτούμενη πληροφορία.

Στο πρόγραμμα που δημιουργήθηκε για τις ανάγκες της υπηρεσίας αυτόματης ανάκτησης συνδεδεμένης δομής, με την python θέλουμε να πετύχουμε επικοινωνία μεταξύ των δύο βάσεων, της εξειδικευμένης βάσης που βρίσκεται τοπικά στον υπολογιστή μας και τη βάση της αντίστοιχης υπηρεσίας που προσφέρει η βιβλιοθήκη του Κογκρέσου.

Από την εξειδικευμένη βάση βλέπουμε αρχικά τι επικεφαλίδες υπάρχουν αποθηκευμένες. Έπειτα ζητάμε από τη βάση της βιβλιοθήκης του Κογκρέσου να μας επιστρέψει όλους του ευρύτερους, στενότερους και συναφείς θεματικούς όρους. Τέλος για κάθε ένα αποτέλεσμα, να αναζητήσει από την εξειδικευμένη βάση ποια από τα αποτελέσματα υπάρχουν. Ακολουθεί μια αφαιρετική περιγραφή του αλγορίθμου:

- Βρες την συνδετική δομή (getSyndeticStructure.py)
- Δημιούργησε ένα Sparql ερώτημα για κάθε ένα μόνιμο προσδιοριστή στην εξειδικευμένη βάση localhost:8000/test (sparql query for distinct URIs in localhost:8000)
- Για κάθε ένα από τα παραπάνω αποτελέσματα δημιούργησε ένα sparql ερώτημα για τους ευρύτερους στενότερους και συναφείς θεματικούς όρους από την βιβλιοθήκη του Κογκρέσου *http://83.212.99.113:8089* (for each sparql results, sparql query for broader narrower and related terms in DB LCSH)
- Για κάθε μία από τις τρεις λίστες (ευρύτερος, πλατύτερος ή συναφής θεματικός όρος) των αποτελεσμάτων που αντιστοιχούν σε ένα μόνιμο προσδιοριστή, βρες αν υπάρχουν τα αποτελέσματα στην εξειδικευμένη τοπική βάση δεδομένων localhost:8000 (for each of this lists BT, NT, RT, find if the items exists in localhost:8000).
- Εκτύπωσε τα αποτελέσματα στην οθόνη (print)

[7] Πρόσβαση: http://di.ionio.gr/~mistral/tp/intprogr/unit1/module1.html#python Ημ. Πρόσβασης: 22/11/2013

ΕΞΥΠΗΡΕΤΗΤΗΣ TWISTED

Το ολοκληρωμένο σύστημα εξυπηρετητή ανοιχτού κώδικα twisted είναι υλοποιημένο σε python και επιτρέπει την ανάπτυξη διαδικτυακών εφαρμογών μεγάλης κλίμακας[8]. Το twisted αποτελείται από πάρα πολλές βιβλιοθήκες[9]. Στο πλαίσιο της εργασίας αυτής αναπτύχθηκε κώδικας προκειμένου να υλοποιηθούν όλες οι επικοινωνίες μεταξύ των τριών επιπέδων της υπηρεσίας. Ο κώδικας αυτός είναι διαθέσιμος στο παράρτημα πηγαίου κώδικα.

Πιο συγκεκριμένα, μέσω του εξυπηρετητή twisted κατασκευάστηκε μία θύρα (TCP port: 8888) η οποία δέχεται κλήσεις HTTP από τον Web Browser. Οι κλήσεις αυτές ουσιαστικά υλοποιούν ένα πρωτόκολλο που βασίζεται σε αιτήσεις GET. Το πρωτόκολλο αυτό εξυπηρετεί δύο βασικές λειτουργίες. Η πρώτη λειτουργία αντιστοιχεί σε ένα σύστημα ελέγχου πληκτρολόγησης (autocomplete control) το οποίο δέχεται την πληκτρολόγιση του χρήστη και επιστρέφει θεματικές επικεφαλίδες που αρχίζουν με τα γράμματα που πληκτρολόγησε ο χρήστης. Η λειτουργικότητα αυτή εξυπηρετείται από την κλάση NativeAutosuggest του αρχείου twisted-xhr.py.

Η δεύτερη λειτουργία αντιστοιχεί στην συγκέντρωση της συνδετικής δομής της θεματικής επικεφαλίδας που επέλεξε ο χρήστης από το σύστημα ελέγχου πληκτρολόγησης που αναφέρθηκε πιο πάνω. Περισσότερες λεπτομέρειες για την παρεχόμενη λειτουργικότητα του εξυπηρετητή twisted, περιγράφονται στο παράρτημα ‘πηγαίος κώδικας’ της εργασίας αυτής.

[8] Πρόσβαση: http://di.ionio.gr/~mistral/tp/intprogr/unit5/module2.html Ημ. Πρόσβασης: 23/11/2013

[9] Πρόσβαση: http://di.ionio.gr/~mistral/tp/intprogr/unit5/module2.html Ημ. Πρόσβασης: 23/11/2013

ΣΥΜΠΕΡΑΣΜΑΤΑ

Ο σύγχρονος Βιβλιοθηκονόμος καλείται να χρησιμοποιήσει όλες τις υπηρεσίες και τα εργαλεία που του προσφέρονται καθώς η τεχνολογία και ο κλάδος της πληροφορικής εξελίσσονται. Δεν χρειάζεται πλέον να χάσει ώρες αναζητώντας θεματικές επικεφαλίδες στους παραδοσιακούς καταλόγους και πιθανές σχέσεις που μπορεί να προκύπτουν ανάμεσα τους.

Η υπηρεσία αυτόματης ανάκτησης συνδεδεμένης δομής που παρουσιάστηκε στην παρούσα μεταπτυχιακή διατριβή, δίνει την δυνατότητα στον βιβλιοθηκονόμο ή ακόμα και στους απλούς χρήστες των ψηφιακών βιβλιοθηκών να αντλήσουν πληροφορίες των ευρύτερων, στενότερων ή συναφών όρων, για κάθε μία τοπική θεματική επικεφαλίδα που αντιστοιχεί σε θεματική επικεφαλίδα του αποθετηρίου της βιβλιοθήκης του Κογκρέσου.

Ο προγραμματιστής από την πλευρά του καλείται να ‘φτιάξει’ τέτοιου είδους υπηρεσίες, ανταποκρινόμενος στις ανάγκες των χρηστών. Με τη χρήση κατάλληλων μεθόδων που του προφέρονται από τον σημερινό σημασιολογικό ιστό μπορεί να επιτύχει το στόχο του.

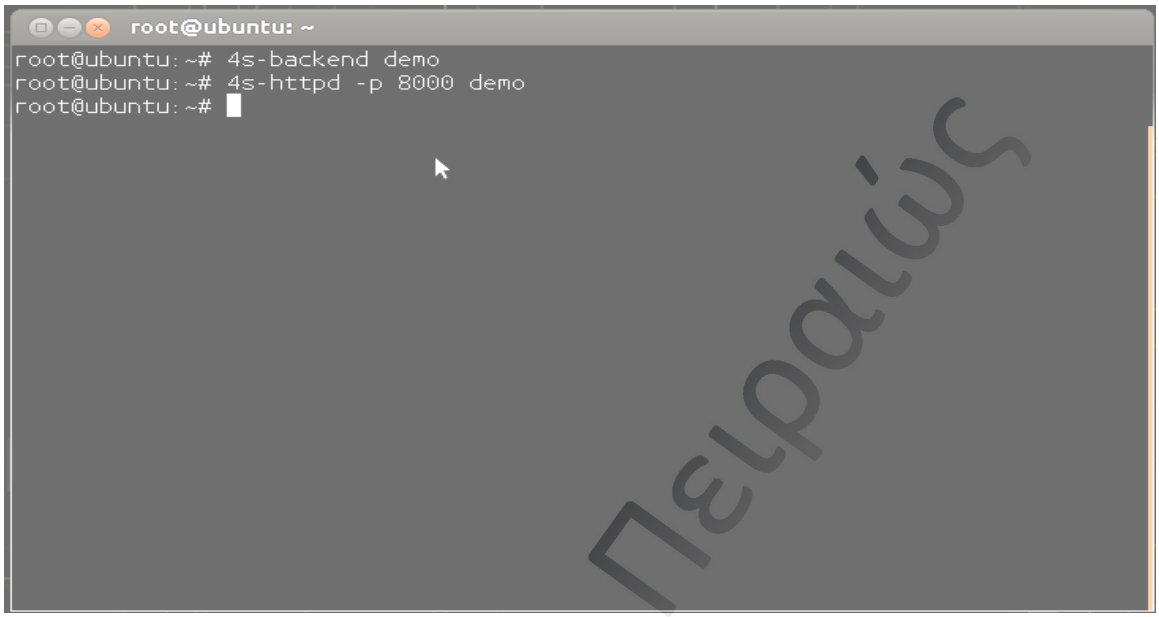
Πλέον δεν χρειάζονται οι μεγάλες βάσεις για την αποθήκευση των δεδομένων σε υπολογιστές με μεγάλη χωρητικότητα. Με τη χρήση των εργαλείων του σημασιολογικού ιστού και τον κατάλληλο προγραμματισμό σε μια γλώσσα εύκολη με υψηλές δυνατότητες (python) γίνονται όλα πιο εύκολα και γρήγορα.

Είναι αναγκαίο να τονίσουμε τη σημασία της υπηρεσίας των ανοικτών συνδεδεμένων δεδομένων, καθώς παρέχει τη δυνατότητα να αποτυπωθούν οι διάφορες σχέσεις ανάμεσα στις θεματικές επικεφαλίδες χωρίς να είναι αποθηκευμένες τοπικά.

Ευελπιστούμε μελλοντικά στην επέκταση των λειτουργιών της υπηρεσίας αυτόματης ανάκτησης συνδεδεμένης δομής, δίνοντας περαιτέρω δυνατότητες στη βιβλιοθηκονομική κοινότητα χρησιμοποιώντας τεχνολογίες του σημασιολογικού ιστού.

ΠΑΡΑΡΤΗΜΑ Α: ΧΡΗΣΙΜΟΠΟΙΟΥΜΕΝΕΣ ΥΠΗΡΕΣΙΕΣ

ΕΙΚΟΝΕΣ ΕΞΥΠΗΡΕΤΗΤΗ 4STORE KB DEMO

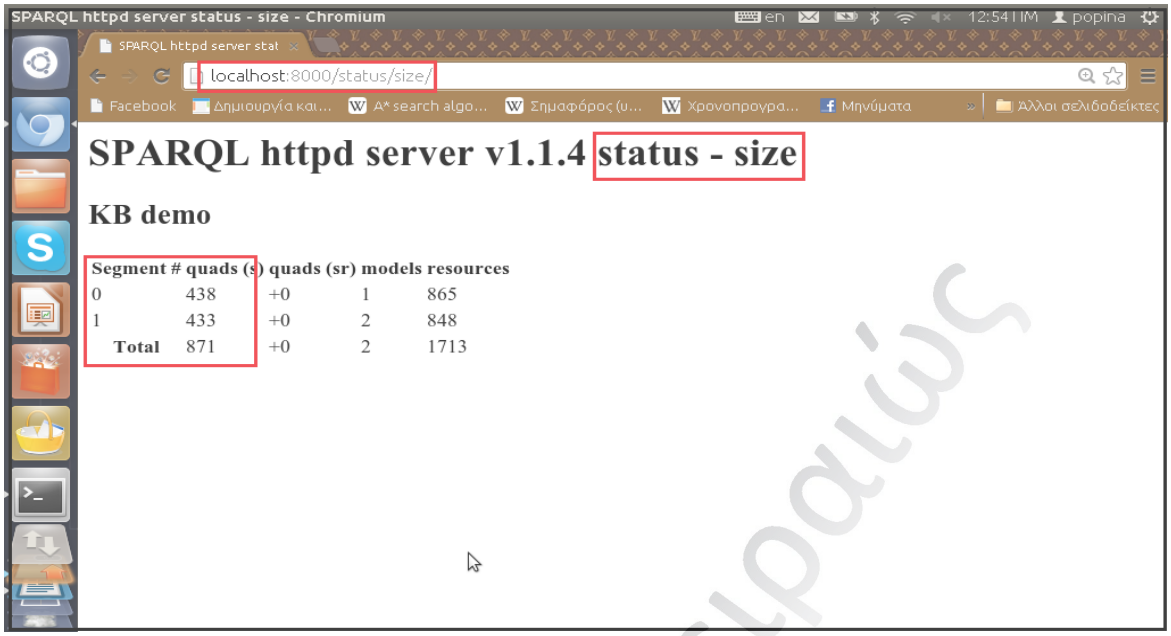


Εικόνα 1 – Άνοιγμα back-end από το τερματικό Linux

Στην εικόνα 1 εμφανίζονται οι εντολές που πρέπει να πληκτρολογήσουμε έτσι ώστε να συνδεθούμε με τον εξυπηρετητή του 4Store. Για να συνδεθούμε ως root πληκτρολογούμε την εντολή *sudo –i* και πληκτρολογούμε τον κωδικό συστήματος (αν υπάρχει).

Εικόνα 2 – Σύνδεση στον τοπικό εξυπηρετητή (localhost:8000/demo)

Ανοίγοντας ένα πρόγραμμα περιήγησης και πληκτρολογώντας στην διεύθυνση *localhost:8000/status* εμφανίζονται οι εξής πληροφορίες: α) Το όνομα του εξυπηρετητή (SPARQL http server v1.1.4 status) και β) η ονομασία της εξειδικευμένης βάσης που έχουμε δημιουργήσει (KB demo), εικόνα 2. Μπορούμε να επιλέξουμε *4store backend size info* για να δούμε το σύνολο από τις N-triples που είναι αποθηκευμένες στην εξειδικευμένη βάση δεδομένων ή να εκτελέσουμε ένα Sparql ερώτημα (execute a test query).

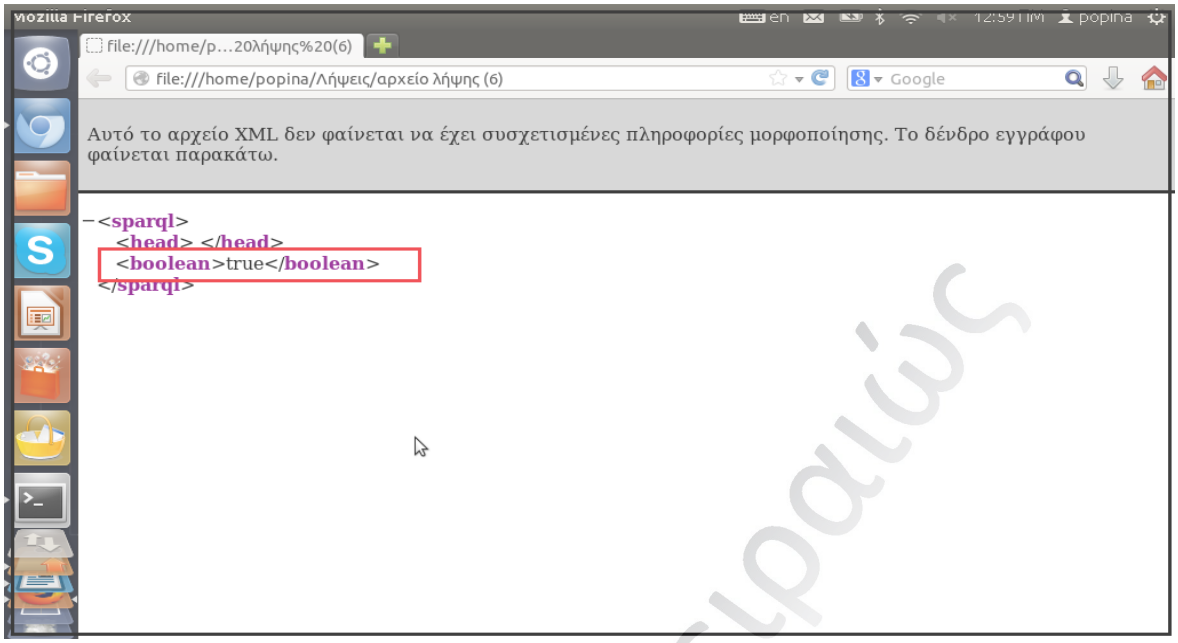


Εικόνα 3 – Συνολικό μέγεθος από τριπλέτες

Καθώς επιλέγουμε *4store backend size info*, ενημερωνόμαστε ότι είναι αποθηκευμένες 871 N-triples όπως εμφανίζεται στην εικόνα 3.

Εικόνα 4 – Εκτελώντας ένα απλό sparql – ερώτημα

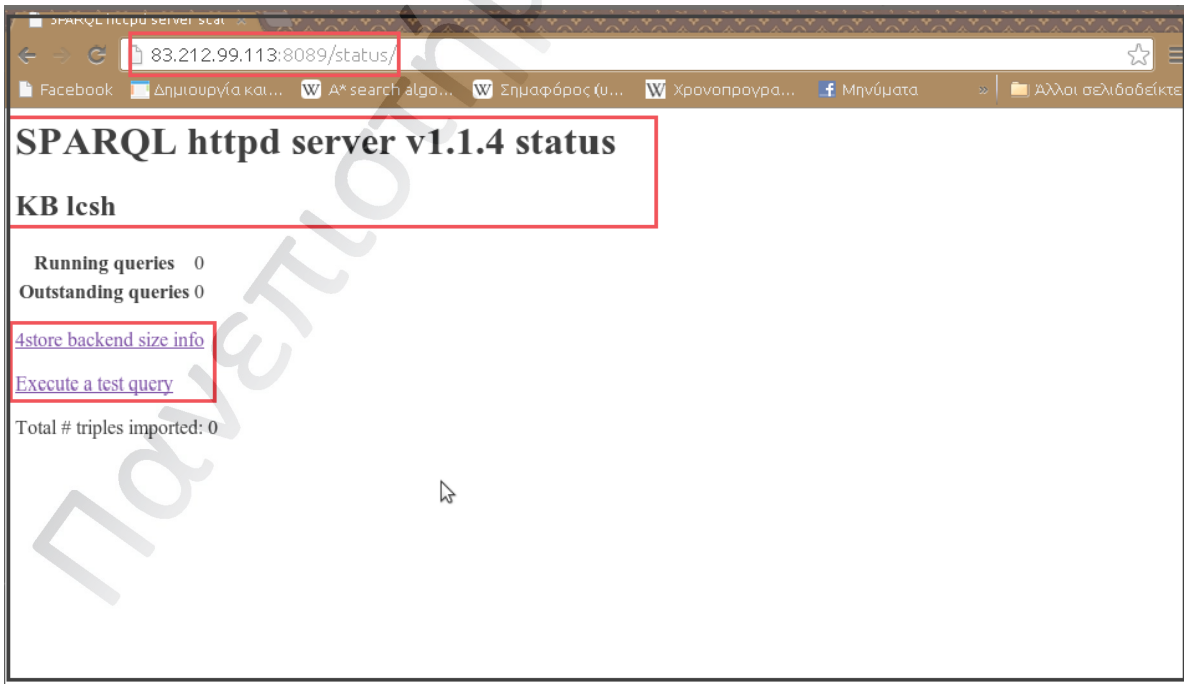
Επιλέγοντας εκτέλεση ενός sparql – ερωτήματος (execute a test query), εμφανίζεται η παραπάνω φόρμα στον εξυπηρετητή. Στο παράδειγμα ζητάμε από τον εξυπηρετητή να μας επιστρέψει για τη θεματική επικεφαλίδα "ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ"@GR με Uri http://id.loc.gov/authorities/sh85006210#concept, αν υπάρχει στην εξειδικευμένη βάση δεδομένων (kb demo). Επιλέγοντας εκτέλεση του Sparql ερωτήματος (Execute) αυτόματα γίνεται λήψη του αποτελέσματος.



Εικόνα 5 – Εμφάνιση αποτελέσματος σε xml

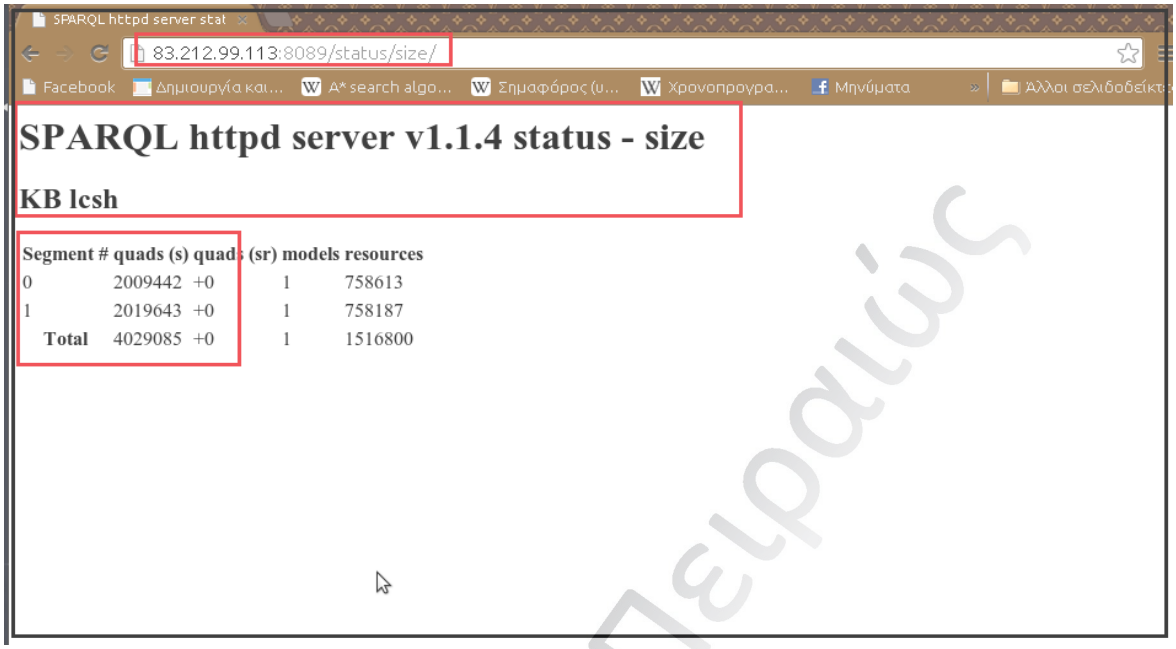
Εκτελώντας το παραπάνω παράδειγμα, το αποτέλεσμα εμφανίζεται αυτόματα στην μορφή XML. Στην εικόνα 5 βλέπουμε την λέξη 'true'. Αυτό σημαίνει πως η θεματική επικεφαλίδα του Sparql ερωτήματος που εκτελέσαμε υπάρχει στην εξειδικευμένη βάση (kb Demo). Αν δεν υπήρχε η θεματική επικεφαλίδα, η απάντηση στην XML θα ήταν 'false'.

ΕΙΚΟΝΕΣ ΕΞΥΠΗΡΕΤΗΤΗ HTTP4STORE LCSH



Εικόνα 6 – Sparql εξυπηρετητής της Βιβλιοθήκης του Κογκρέσου

Ανοίγοντας ένα πρόγραμμα περιήγησης πληκτρολογώντας την διεύθυνση *83.212.99.113:8089*, με τη χρήση του HTTP Sparql εξυπηρετητή μπορούμε να συνδεθούμε στο αποθετήριο της βιβλιοθήκης του Κογκρέσου (εικόνα 6).

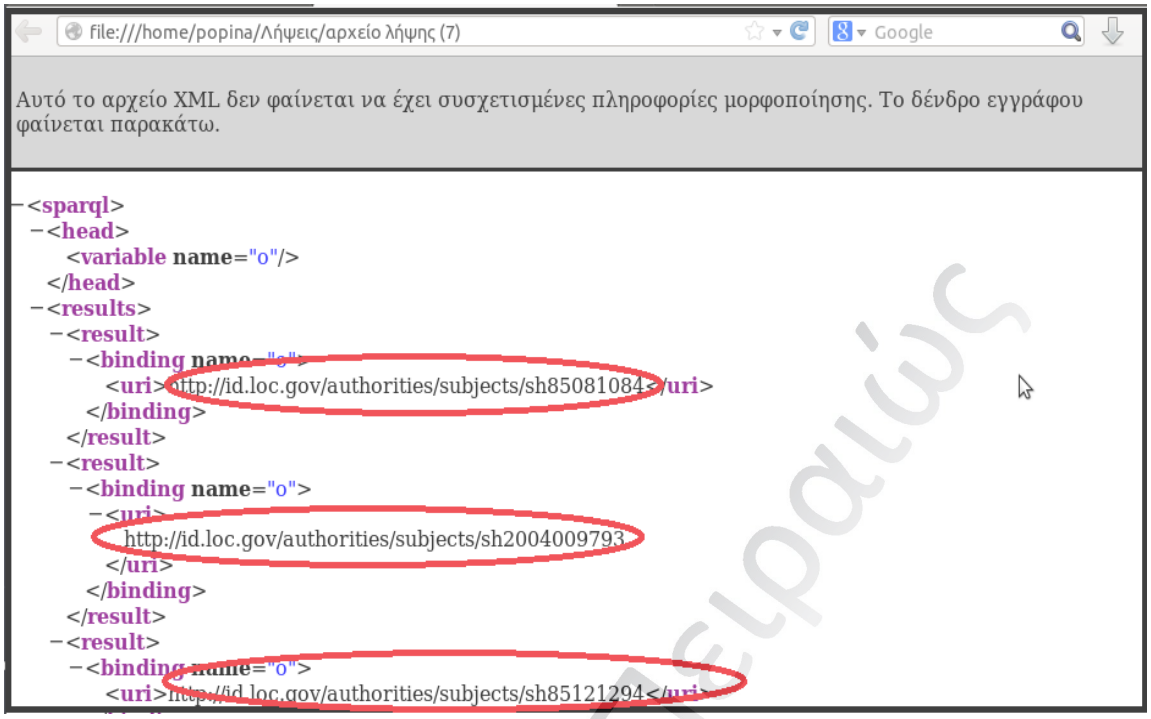


Εικόνα 7 – Συνολικός μέγεθος από τριπλέτες της Βιβλιοθήκης του Κογκρέσου

Επιλέγοντας *4store size info* βλέπουμε ότι είναι αποθηκευμένες στον εξυπηρετητή της βιβλιοθήκης του Κογκρέσου 4.029.085 N-triples.

Εικόνα 8 – Εκτελώντας ένα sparql – ερώτημα στον εξυπηρετητή της Βιβλιοθήκης του Κογκρέσου

Στο παραπάνω Sparql ερώτημα ζητάμε να μας εμφανίσει όλους τους στενότερους όρους που μπορεί να έχει η θεματική επικεφαλίδα "ΥΔΑΤΟΚΑΛΛΙΕΡΓΕΙΑ"@GR που το ενιαίο αναγνωριστικό πόρου αυτής είναι: http://id.loc.gov/authorities/sh85006210#concept.



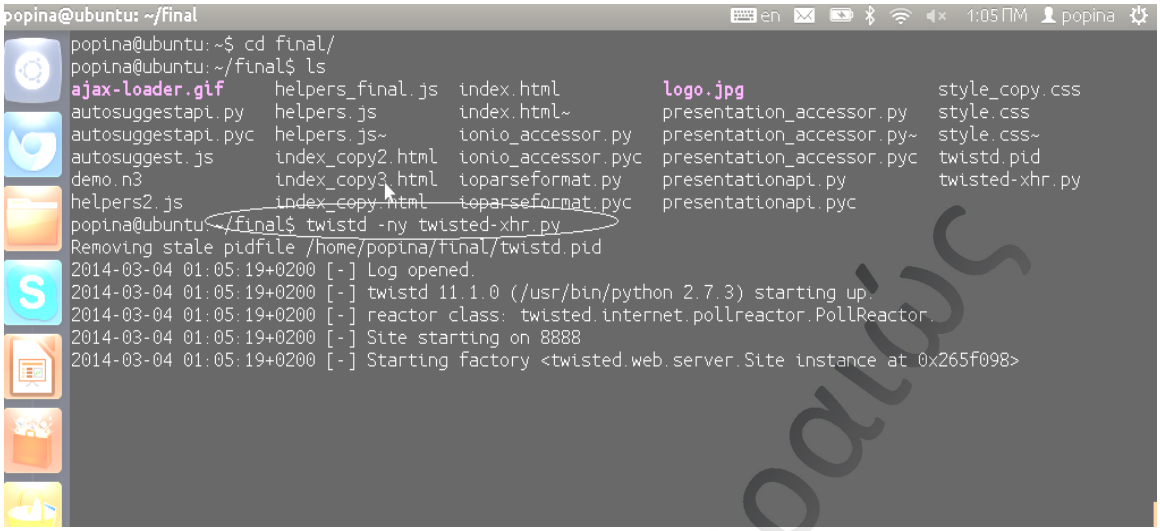
Εικόνα 9 – Αποτελέσματα sparql –ερωτήματος της Βιβλιοθήκης του Κογκρέσου σε xml

Πατώντας εκτέλεση του Sparql ερωτήματος αυτόματα γίνεται λήψη των αποτελεσμάτων. Η θεματική επικεφαλίδα έχει αρκετούς στενότερους όρους.

ΕΙΚΟΝΕΣ ΥΠΗΡΕΣΙΑΣ ΘΕΜΑΤΙΚΗΣ ΑΝΑΖΗΤΗΣΗΣ ΕΠΙΚΕΦΑΛΙΔΩΝ

Έχοντας ανοιχτά τα back-end ως root, έχουμε δημιουργήσει ένα φάκελο ως διαχειριστές στο τερματικό των Linux με αρχεία του twisted εξυπηρετητή σε γλώσσα python. Για να ανοίξουμε την υπηρεσία αυτόματης ανάκτησης συνδεδεμένης δομής πρέπει να ανοίξουμε τον εξυπηρετητή από το τερματικό (ως διαχειριστές) να πληκτρολογήσουμε την εντολή *twistd -ny twisted-xhr.py.*

ΕΙΚΟΝΕΣ ΥΠΗΡΕΣΙΑΣ ΔΙΑΧΕΙΡΙΣΗΣ ΘΕΜΑΤΙΚΩΝ ΕΠΙΚΕΦΑΛΙΔΩΝ



Εικόνα 10 – Ξεκινώντας τον twisted εξυπηρετητή από το τερματικό

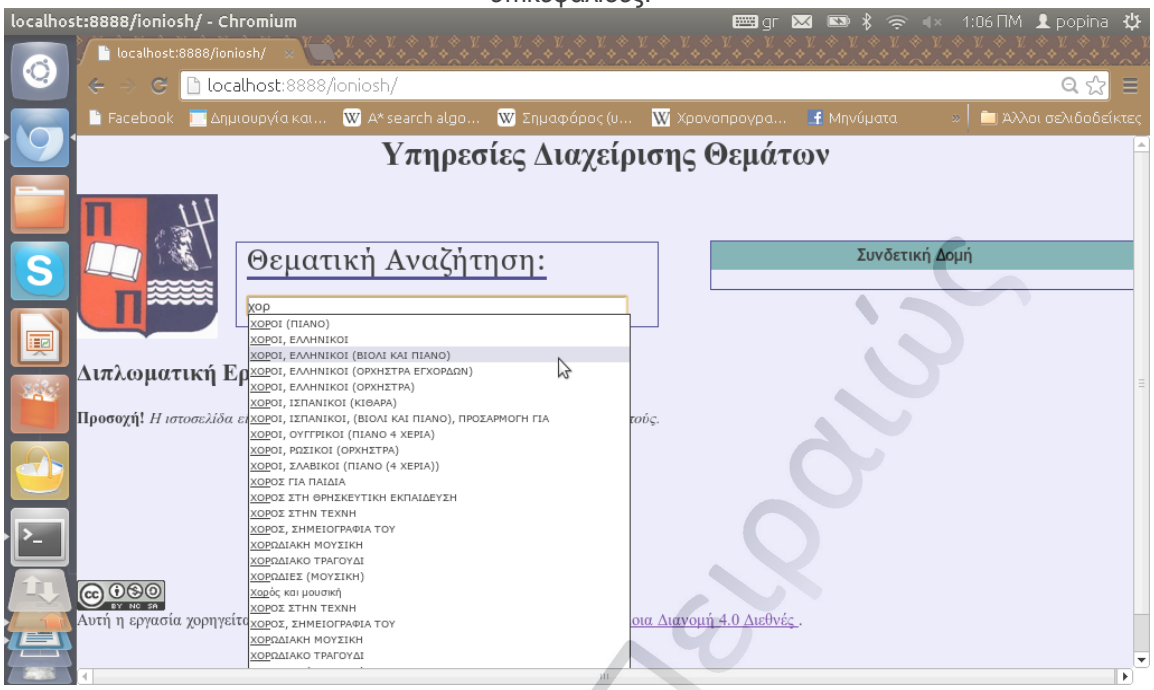
Στην εικόνα 10 στην πρώτη γραμμή με την εντολή *cd final/* μπαίνουμε στον κατάλογο με τα αρχεία. Στη δεύτερη γραμμή μέσα στον κατάλογο final πληκτρολογούμε $ ls για να δούμε τα αρχεία που βρίσκονται μέσα στο final (autosuggestapi.py, helpers.js, index.html κτλ.)

Με την εντολή *twistd -ny twisted-xhr.py* βλέπουμε ότι ξεκίνησε ο εξυπηρετητής την ακριβή ημερομηνία και ώρα. Βλέπουμε επίσης σε ποια θύρα τρέχει ο εξυπηρετητής, στην port:8888 (site starting on 8888).

Εικόνα 11 – Υπηρεσία Διαχείρισης Θεματικών Επικεφαλίδων

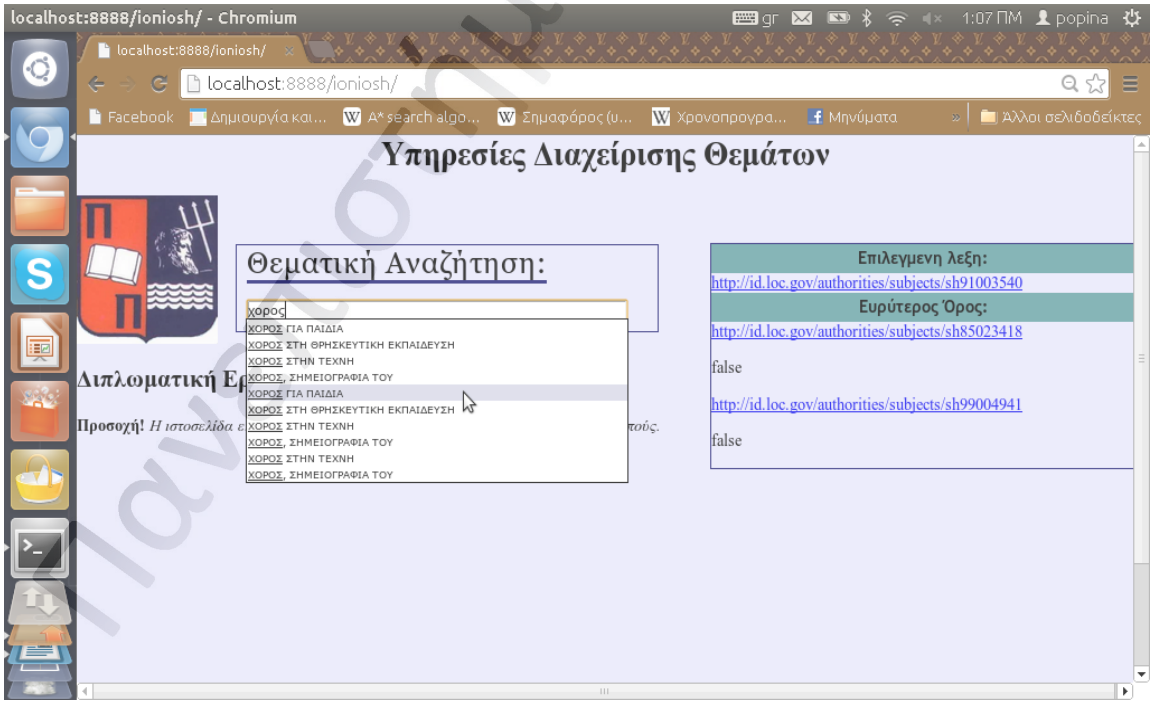
Ανοίγοντας ένα πρόγραμμα περιήγησης πληκτρολογώντας τη διεύθυνση *localhost:8888/ionioshi/* εμφανίζεται η αρχική σελίδα της υπηρεσίας. Η υπηρεσία διαθέτει ένα κουτί κειμένου (text box) όπου ο χρήστης έχει τη δυνατότητα να αναζητήσει θεματικές

επικεφαλίδες.



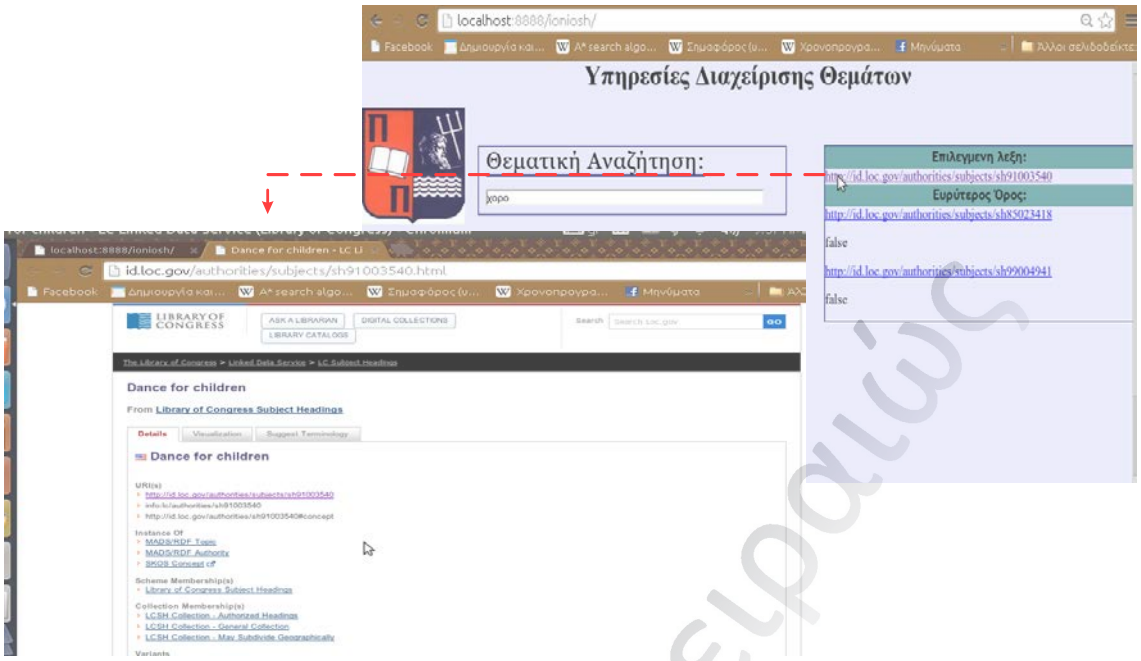
Εικόνα 12 – Αναζήτηση λέξεων/θεματικών επικεφαλίδων

Στην εικόνα 12 εμφανίζεται μια λίστα από θεματικές επικεφαλίδες που ξεκινάνε από ‘χορ’. Η υπηρεσία αναζητά από την εξειδικευμένη βάση δεδομένων (Kb Demo) όλους τους προτιμώμενους όρους, εμφανίζοντας στον χρήστη όλα τα αποτελέσματα που υπάρχουν από ‘χορ’.



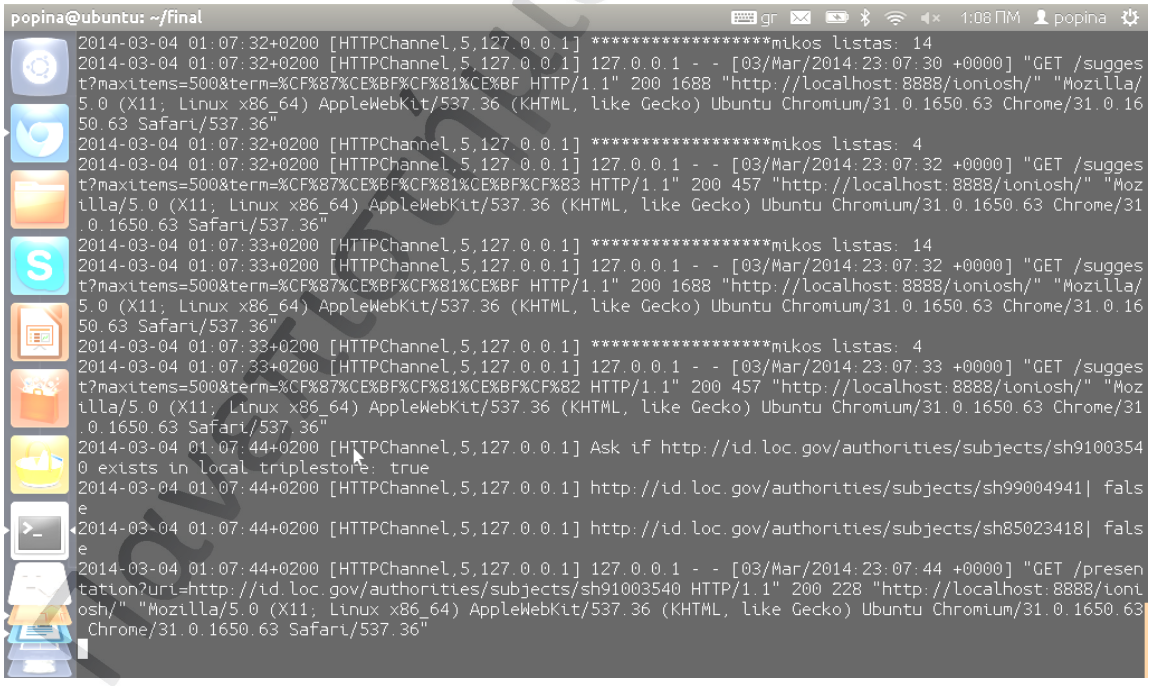
Εικόνα 13 – Εμφάνιση αποτελεσμάτων θεματικής επικεφαλίδας

Κάνοντας κλίκ πάνω σε μια θεματική επικεφαλίδα όπως ‘ΧΟΡΟΣ ΓΙΑ ΠΑΙΔΙΑ’, στο δεξιό τμήμα βλέπουμε τα αποτελέσματα. Τα αποτελέσματα προκύπτουν από την αναζήτηση στην αντίστοιχη υπηρεσία της βιβλιοθήκης του Κογκρέσου (Kb LCSH).



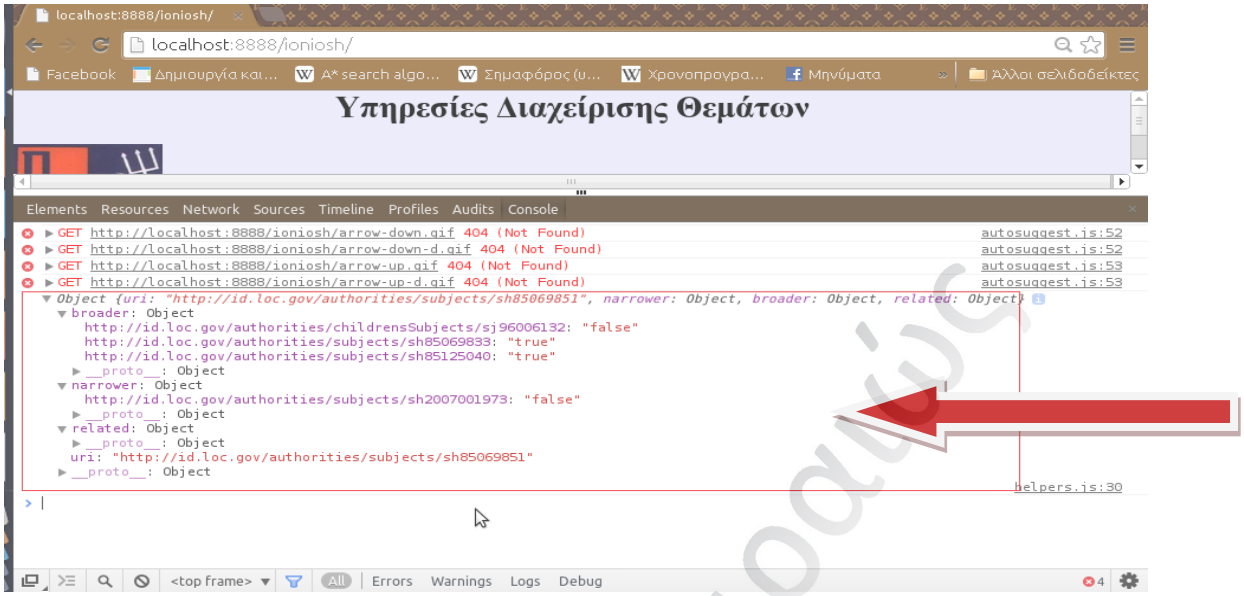
Εικόνα 14 – Πατώντας πάνω στο Uri της επιλεγμένης λέξης

Η υπηρεσία δίνει τη δυνατότητα στο χρήστη κάνοντας κλίκ πάνω σε κάποιο αποτέλεσμα στον πίνακα της Συνδετικής Δομής, να ανοίξει αυτόματα σε νέα καρτέλα τη σελίδα της βιβλιοθήκης του Κογκρέσου.



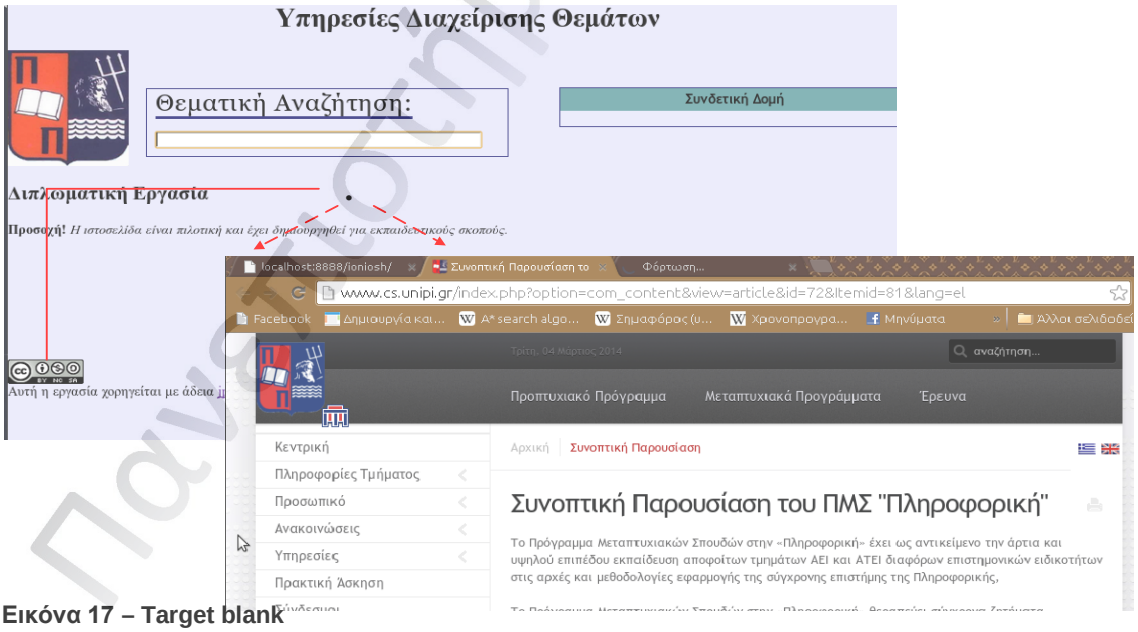
Εικόνα 15 – Φωτογραφία τερματικού καθώς λειτουργεί η υπηρεσία

Καθώς η υπηρεσία είναι σε λειτουργιά, όσο ο χρήστης κάνει διάφορες αναζητήσεις, στο τερματικό καταγράφονται όλες οι πληροφορίες. Η πληροφορία με πιο πρόγραμμα περιήγησης έχει συνδεθεί ο χρήστης, τα αποτελέσματα των θεματικών επικεφαλίδων που πληκτρολογεί ο χρήστης, ακόμα και για το ποια Sparql ερωτήματα χρησιμοποιεί η υπηρεσία κάθε φορά που ο χρήστης αναζητεί μια θεματική επικεφαλίδα.



Εικόνα 16- Εργαλεία για Προγραμματιστές από το πρόγραμμα περιήγησης

Όλα τα προγράμματα περιήγησης στους ιστότοπους έχουν διάφορες λειτουργίες και ρυθμίσεις. Στα παραδείγματά μας χρησιμοποιούμε το google chrome. Βρισκόμενοι στην σελίδα που της υπηρεσίας διαχείρισης θεματικών επικεφαλίδων, από την προσαρμογή ελέγχου του google chrome , επιλέγουμε Εργαλεία → Εργαλεία για Προγραμματιστές. Το πρόγραμμα περιήγησης μας δίνει τη δυνατότητα να δούμε τα αποτελέσματα από την επιλογή του χρήστη. Στο συγκεκριμένο παράδειγμα έχουμε επιλέξει την θεματική επικεφαλίδα "ΤΖΑΖ, ΜΟΥΣΙΚΗ ΓΙΑ" με URI http://id.loc.gov/authorities/sh85069851#concept. Στην κονσόλα εμφανίζονται τα αποτελέσματα των ευρύτερων, στενότερων και συναφών όρων (BT, NT, RT). (Εικόνα 16)



Εικόνα 17 – Target blank

Πατώντας πάνω στα δικαιώματα της υπηρεσίας (creative commons), η υπηρεσία παραπέμπει σε νέα καρτέλα τη σελίδα του μεταπτυχιακού του Πανεπιστημίου Πειραιά.

ΠΑΡΑΡΤΗΜΑ Β: ΠΗΓΑΙΟΣ ΚΩΔΙΚΑΣ

twisted-xhr.py

```
    # -*- coding: utf-8 -*-
    # this should be run with the twistd (e.g. "twistd -ny twisted-xhr-nh.py" or only -y to
daemonize)
    # this is a clone of original "twisted-xhr.py"



    from twisted.application import internet,service
    from twisted.web import server,resource
    from twisted.web.static import File
    from twisted.internet import reactor,defer

    # modules of services provided
    import autosuggestapi
    import presentationapi

    # (hardwired) definitions of resources
    class Root(resource.Resource):        # the / resource class
        isLeaf = False

    class PresentSuggestion(resource.Resource):     # servicepoint resource for native
autosuggest requests
        isLeaf = True                    # this is a 'named' service (kburi is explicit, not in
request)

        def __init__(self):
            """ used only to store the global kbmanager instance """
            """ and the resource name of kb to used for autosuggest """
            resource.Resource.__init__(self)

        def render_GET(self,request):
            """ function called when servicing GET requests """
            """ for native autosuggest api. Returns immediatelly, real service """
            """ should be in a callback of autosuggestapi service module"""

            # create instance of service handler
            sh = presentationapi.NativeService(request)
            # call handler - this returns immediatelly!
            sh.service_handle_request()
            # signal 'not done' - real work will be in a callback
            return server.NOT_DONE_YET      # more of response to follow
```

```
    class NativeAutosuggest(resource.Resource):           # servicepoint resource for native
autosuggest requests
        isLeaf = True                                     # this is a 'named' service (kburi is explicit,
not in request)

        def __init__(self):
            """ used only to store the global kbmanager instance """
            """ and the resource name of kb to used for autosuggest """
            resource.Resource.__init__(self)

        def render_GET(self,request):
            """ function called when servicing GET requests """
            """ for native autosuggest api. Returns immediatelly, real service """
            """ should be in a callback of autosuggestapi service module"""

            # create instance of service handler
            sh = autosuggestapi.NativeService(request)
            # call handler - this returns immediatelly!
            sh.service_handle_request()
            # signal 'not done' - real work will be in a callback
            return server.NOT_DONE_YET # more of response to follow


    class DummyLogger(resource.Resource):     # servicepoint resource for dummy logging -
dump any GET request
                                              # you want to be logged here, discard returned value
        isLeaf = True

        def render_GET(self,request):
            """ function called when servicing GET requests, returns a simple text plain
reply """
            request.setHeader('Content-type','text/plain')
            return "ok"

    # main app part, to start the http server
    # global variable MUST be called exactly "application"!!!
    application = service.Application('ioniosh')

    # build the resources tree
    root = Root()
    root.putChild("ioniosh",File("./"))        # to serve normal files (GET) under /ranked
    root.putChild("suggest",NativeAutosuggest()) # to serve native autosuggest requests
(GET) for registry
    root.putChild("presentation",PresentSuggestion())
```

```
    # here we connect a "Site" (subclass of "twistewd.web.http.HTTPFactory") to a "resource"
    site = server.Site(root)
    # start a (reactor based) web server listening
    internet.TCPServer(8888,site).setServiceParent(service.IServiceCollection(application))
```

presentetionapi.py

```
# -*- coding: utf-8 -*-
# this should be loaded only as a module

# module that provides generic autosuggest api servicing
# connects to kb accessors for the real data

from twisted.internet import defer

import ioparseformat
import presentation_accessor

class NativeService:
	""" class providing functions for native autosuggest servicing """

	def __init__(self,request):
		self.request = request


	def service_handle_request(self):
		""" entrypoint for GET requests to native autosuggest api """

		# init an instance of input parser to extract GET input data
		ip = ioparseformat.InputParser()

		# extract get info
		idict = ip.parseUnstructuredGetData(self.request)	#
		uri = idict.get('uri')
		# get the accessor object for this service
		ac = presentation_accessor.presentationAccessor()
		# call autosuggest method of accessor - this returns a tuple: a defered and a string!
		defer = ac.get_syndeticStructure(uri=uri)
			# add callback to do the output when finishing
		defer.addCallback(self.output_callback)

		# NOTE: nothing returned here - our caller (http server) does not expect anything
now

	def output_callback(self,result):
		""" sends autosuggest response, result may be empty list. autochange is true if
```

```
results come from alphabetical despite the different radiobutton"""
            # init an instance of output formatter to send output
            of = ioparseformat. Output Formatter()
            # send output
            of.deliverJsonData  (self.request,result)
```

presentation_accessor.py

```
# -*- coding: utf-8 -*-
# should run only as module

# this module provides accessor methods for lcsh KB

from twisted.internet import defer
import re
import requests
import xml.etree.ElementTree as et
import json
import collections

class presentationAccessor:

        def __init__(self):
                self.host= 'http://localhost:8000/'
                self.host2= 'http://83.212.99.113:8089/'
                # TBD
                pass

        def get_syndeticStructure(self,uri):
                jsonresult = collections.OrderedDict()
#               host= 'http://localhost:8002/'
#               host2= 'http://localhost:8089/'
                query="ask where { <" + uri + "> ?p ?o }"
                data= {'query':query,'output':'xml'}
                request=requests.post(self.host +"sparql/",data=data)
                if request.status_code != requests.codes.ok:   # something went wrong
                        print "error: "+ str(request.status_code)

                else:
                        result=et.fromstring(request.text)
```

```
                        print "Ask if "+ uri +" exists in local triplestore: " + result[1].text
                        if result[1].text:
                                jsonresult["uri"]=uri
                                jsonresult["narrower"]= self.getData(uri, "narrower")
                                jsonresult["broader"]=self.getData(uri, "broader")
                                jsonresult["related"]=self.getData(uri, "related")

                # as we don't have anything else to do here, we return a succeded defer
                sucdefer = defer.succeed(jsonresult)      # result will be passed to any
callback attached
                # return the defered
                return sucdefer

        def getData(self,uri,relation):
                query='prefix Skos:<http://www.w3.org/2004/02/Skos/core#> select ?o where
{ <' + uri + '> Skos:'+ relation +' ?o.}'
            data= {'query':query,'output':'xml'}
            request=requests.post(self.host2 +"sparql/",data=data)
            if request.status_code != requests.codes.ok:   # something went wrong
                print "error: "+ str(request.status_code)
            else:
                result=et.fromstring(request.text)
                jsonrelation=dict()
                for child in result[1]: #gia ta paidia tou <results>
                  asklocal = "ask where { <"+ child[0][0].text  +"> ?p ?o }"
                    datalocal= {'query':asklocal,'output':'xml'}
                    asklocalrequest =requests.post(self.host +"sparql/",data=datalocal)
                    if asklocalrequest.status_code != requests.codes.ok:   # something went
wrong
                        print "error: "+ str(asklocalrequest.status_code)
                    else:
                        asklocalresult=et.fromstring(asklocalrequest.text)
                         jsonrelation[child[0][0].text]=asklocalresult[1].text
                         print child[0][0].text + "| " + asklocalresult[1].text
                return jsonrelation
```

ioparseformat.py

```
# -*- coding: utf-8 -*-
# this should be loaded only as a module

# module that provides various classes and methods to parse request input data
# and format response output data

import re
import json

class InputParser:
	""" a class that provides various methods to parse and extract """
	""" GET/POST request input data """

	def __init__(self):
		# generic error flag
		self.generror = False
		#generic error msg
		self.generrmsg = ''

	def parseUnstructuredGetData(self,request):
		""" a simple class to parse GET params as unstructured data """
		""" (that is, as a flat key-value pair dict) which is returned """

		# the dict to be returned
		retdict = {}
		# simply pass all request args (which are actually in a dict!) to the dict
returned
		for k,v in request.args.iteritems():
			retdict[k] = v[0]  # twisted gives GET args in {key:[value],...} format!
		# return the new (flat!) dict
		return retdict

class OutputFormatter:
	""" a class that provides methods to output data in various formats """

	def xmlsafe(self,istr):
		""" utility function to sanitize xml strings by replacing & and < """
		return istr.replace("&","&amp;").replace("<","&lt;")

	def xmlquotesafe(self,istr):
		""" utility function to sanitize xml strings by replacing &, " and < """
		return istr.replace("&","&amp;").replace("<","&lt;").replace("","&quot;")
```

```
        def formatXMLAutosuggestNative(self,request,itemlist):
                """ outputs a list of items as an XML response """
                """ list contains tuples of (item-id,item-weight,item-label, item-node) """
                """ and output is of the form <listdata>item-label,item-id|item-label,item-
id|..</listdata> """
                """ labels and ids must be already utf-8 encoded """
                sepr = ','
#               for itemid,itemweight,itemlabel,itemnode in itemlist:
#               itemnodeprevious = ''
#               counter = 0
#               ip = InputParser()
#               getdict = ip.parseUnstructuredGetData(request)
#               maxitems = int(getdict.get('maxitems'))
                outstr = '['
          request.setHeader('Content-type','application/json')
          for item in itemlist:
                listkv = item.split("|")
                outstr += '{"label":"' + listkv[1] + '","value":"' + listkv[0] + '"},'
          outstr = outstr.rstrip(',')
          outstr += ']'
          request.write(outstr.encode('utf-8'))
          request.finish()

        def deliverJsonData(self,request,response):
                """delivers json results to gui"""
                request.setHeader('Content-type','text/plain')
                #print str(response)
                request.write(json.dumps(response))
                request.finish()

    def deliverPresentationData(self,request,jsonitem):
          """delivers db results to helper.js for further manipulation and post to gui"""
          #if (len(itemlist)!=0): outstr = itemlist.pop(0)
          request.setHeader('Content-type','application/json')
          request.write(jsonitem.encode('utf-8'))
          request.finish()
```

ionio_accessor.py

```
# -*- coding: utf-8 -*-
# should run only as module

# this module provides accessor methods for lcsh KB

from twisted.internet import defer
import re
import requests
import xml.etree.ElementTree as et

class ionioAccessor:

	def __init__(self):
		# TBD
		pass

	def get_suggestions(self,term,maxitems):
		""" returns True and a sorted list of (id,label,weight) tuples that match search term """
		""" or False and a (errorcode,error str) tuple """
		host ='http://localhost:8000/'
		query = ''
		data = ''
		result = []
		# here we ignore maxitems, but term must be present
		if term==None or term=='':
			result = []		# NOTE: here on error we return empty list - error messages have no meaning here
		else:
			# transform search term for search
			# print "prin:" + term + "----------------"
			"""term = term.decode("utf-8")	# convert to unicode
			term = term.lower()	# this may not work for some non-ascii combinations
			# remove other non text chars
			term2 = ''
			for c in term:
				if c.isalnum(): term2 += c		# unicode support here?
			if term2!='': term = term2
			term = term.encode("utf-8")"""

			query = 'SELECT DISTINCT * WHERE {{{ ?subject1 <http://www.w3.org/2004/02/Skos/core#prefLabel> ?label.} UNION { ?subject1
```

```
<http://www.w3.org/2004/02/Skos/core#altLabel> ?label.}} FILTER (regex(?label, "^' + term +
'(.*)", "i")). } ORDER BY ?label LIMIT 50'
                        data = {'query':query,'output':'xml'}
                        request = requests.post(host+"sparql/", data=data)
                        if request.status_code != requests.codes.ok:
                                """there has been an error, so we return an empty list"""
                                print  "ERROR: " + str(request.status_code)
                        else:
                                response = et.fromstring(request.text.encode('utf-8'))
                                for child in response[1]:
                                        result.append(child[0][0].text + '|' + child[1][0].text)
                                        uri = response[1][0][0][0].text
                                        label = response[1][0][1][0].text
                                print "******************mikos listas: " + str(len(result))
                # as we don't have anything else to do here, we return a succeded defer
                sucdefer = defer.succeed(result)# result will be passed to any callback
attached
                # return the defered
                return sucdefer
```

helpers.js

```
    var app_vars = {server:'',thispageurl:'',provenance:''};
    var counter = 0;
    var nodes=0;
 var rexp = /(.*)\/ionioshV/;          // for this to work, the redirector addr must NOT contain
"/ioniosh/"
    var rexr = rexp.exec(location.href);
    if (rexr.length>0) {
          app_vars.server = rexr[1];
      // this doc's URL *excluding any search param
        app_vars.thispageurl = rexr[1] + "/presentation";
    }
    function loadContent(id, thema) {
          document.getElementById("placeholder1").innerHTML = "";
          sendthema(id);
    //    var paidi1 = document.createElement('p');
    //    paidi1.innerHTML = thema;
    //    var paidi1title = document.createElement('h4');
    //    paidi1title.innerHTML = 'Επιλεγμένη θεματική επικεφαλίδα:';
    //    document.getElementById("placeholder1").innerHTML =
'<h3>Request/Response<img src="ajax-loader.gif" class="gif" id="registrygif1"></h3>';
    //    document.getElementById("placeholder1").appendChild(paidi1title);
    //    document.getElementById("placeholder1").appendChild(paidi1);
    }
```

```
function sendthema(id) {
  var xhr = new XMLHttpRequest();
  xhr.open('GET', app_vars.thispageurl + '?uri=' + id, true);
  xhr.responseType = 'text';
  xhr.onload = function(e) {
    if (this.status == 200) {
      var results = JSON.parse(this.response);
      console.log(results);
      //alert(results);
      //alert(Object.keys(results).length);
      alert("Έχετε επιλέξει το παρακάτω URI: " +results["uri"]);
      //addelm('Επιλεγμένη λέξη:',results.uri);
      if(results["uri"]){
            var elmtitle1;
            var label1="Επιλεγμενη λεξη:";
            elmtitle1 = document.createElement('h4');
            elmtitle1.innerHTML = label1;
            document.getElementById("placeholder1").appendChild(elmtitle1);
      var newLink1=document.createElement('a');    //it works yeaaaaaahhhh!!!! :))))
       newLink1.setAttribute('href',results["uri"]);
       newLink1.setAttribute('target','_blank');
      // - use 'className' to create the equivalent of class="contrast"
      //newLink1.className='contrast1';
        var linkText1=document.createTextNode(results["uri"]);
      newLink1.appendChild(linkText1);
      document.getElementById('placeholder1').appendChild(newLink1);
            //var epilegmeni = document.createElement('p');
            //epilegmeni.innerHTML = results.uri;
            //document.getElementById("placeholder1").appendChild(epilegmeni);
      }

      if(Object.keys(results["broader"]).length>0){
            var elmtitle2;
            var label2="Ευρύτερος Όρος:";
            elmtitle2 = document.createElement('h4');
            elmtitle2.innerHTML = label2;
            document.getElementById("placeholder1").appendChild(elmtitle2);
            $.each(results["broader"], function(key, value){
            //.alert("BROADER    key: " + key + ", value: " + value);
            addelm(key,value);
      });

      }
      if(Object.keys(results["narrower"]).length>0){
            var elmtitle3;
            var label3="Στενότερος Όρος:";
```

```
                elmtitle3 = document.createElement('h4');
                elmtitle3.innerHTML = label3;
                document.getElementById("placeholder1").appendChild(elmtitle3);
                $.each(results["narrower"], function(key, value){
                //alert("RELATED    key: " + key + ", value: " + value);
                addelm(key,value);
        });

        }
        if(Object.keys(results["related"]).length>0){
                var elmtitle4;
                var label4="Συναφής Όρος:";
                elmtitle4 = document.createElement('h4');
                elmtitle4.innerHTML = label4;
                document.getElementById("placeholder1").appendChild(elmtitle4);
                $.each(results["related"], function(key, value){
                //alert("NARROWER   key: " + key + ", value: " + value);
                addelm(key,value);
        });

        }

    }
  };
  xhr.send();
}

function addelm(key,value) {
         var newLink=document.createElement('a');
        newLink.setAttribute('href',key);
        // - use 'className' to create the equivalent of class="contrast"
        newLink.className='contrast';
          var linkText=document.createTextNode(key);
        newLink.setAttribute('target','_blank');
        newLink.appendChild(linkText);
        document.getElementById('placeholder1').appendChild(newLink);
        //var elm = document.createElement('p');
        var elm2 = document.createElement('p');
        //elm.innerHTML = key;
        elm2.innerHTML = value;
        //document.getElementById("placeholder1").appendChild(elm);
        document.getElementById("placeholder1").appendChild(elm2);

}
```

index.html

```
<html>
<head>
<h1 align="center">Υπηρεσίες Διαχείρισης Θεμάτων</h1>
<meta http-equiv="Content-type" content="text/html; charset=utf-8" />
<script type="text/javascript" src="autosuggest.js"></script>
<script type="text/javascript" src="helpers.js"></script>
<script type="text/javascript" src="http://code.jquery.com/jquery-1.10.2.min.js"></script>
<link rel="stylesheet" type="text/css" href="style.css"></link>
</head>
<body onload="init()">
<img src="logo.jpg" width="140" height="142">
<h2 align="left">Διπλωματική Εργασία</h2>
<p><b>Προσοχή!</b><i> Η ιστοσελίδα είναι πιλοτική και έχει δημιουργηθεί για
εκπαιδευτικούς σκοπούς.</i></p>
<!--<h3 align="left">ΠΜΣ Πληροφορικής</h3>-->
<div id='placeholder0'>
<a class="title">Θεματική Αναζήτηση:</a><br/><br/>
<!--<input type="radio" name="radioa" id="radio1" checked="true">backlink ranked</input>
<input type="radio" name="radioa" id="radio2">alphabetically ranked</input>
<br/><br/>-->
<input id="tb" title="" type="text" style="overflow: auto; height: 20px; width:95%; opacity:
0.999999;" autocomplete="off"/><img src="ajax-loader.gif" class="gif"
id="autosuggestgif"/></input>
</div>
<div id='placeholder1' class="placeholder1" href="key"target="_blank">
  <h4>Συνδετική Δομή<img src="ajax-loader.gif" class="gif" id="registrygif1"/></h4></br>


  </div>
<!--<div div="terms">
  <center><a
href="http://www.cs.unipi.gr/index.php?option=com_content&view=article&id=72&Itemid=81
&lang=el"> &copy; 2013 jpap/kankall  Terms & Conditions</a></center>
</div>-->
<a rel="license"
href="http://www.cs.unipi.gr/index.php?option=com_content&view=article&id=72&Itemid=81
&lang=el" target="_blank"><img alt="Άδεια Creative Commons" style="border-width:0"
src="http://i.creativecommons.org/l/by-nc-sa/4.0/88x31.png" target="_blank"/></a><br
/>Αυτή η εργασία χορηγείται με άδεια <a rel="license"
href="http://creativecommons.org/licenses/by-nc-sa/4.0/deed.el" target="_blank">jpap &
kankall - Μη Εμπορική Χρήση - Παρόμοια Διανομή 4.0 Διεθνές </a>.



</body></html>
```

```
<!-- select node, count(node) from label_index group by node order by count(node) desc; -->
```

style.css

```
style.css
body {
margin:0;
padding:0;
height:100%;
background:#E6E6FA;
}
img.gif {
float:right;
overflow:auto;
visibility:hidden;
}
a.title {
font: 1.8em Georgia, Times, serif;
text-align: center;
letter-spacing: 1px;
border-color: #191970;
border-bottom-style: solid;
}
div {
border-width: 1px;
border-color: #191970;
border-style: solid;
}
div#placeholder0 {
position: relative;
top: -200px;
left: 15%;
width: 398px;
padding-left:10px;
padding-right:10px;
padding-bottom:10px;
}
div#placeholder1 {
position: relative;
top: -285px;
left: 60%;
width: 420px;
}
h3 {
```

```
background:#8BB381;
text-align: left;
margin:0;
font: bold 1em/1.8em "Century Gothic","Trebuchet MS",Arial,Helvetica,sans-serif;
}
h4{
background:#5F9EA0;
text-align: center;
margin:0;
font: bold 1em/1.8em "Century Gothic","Trebuchet MS",Arial,Helvetica,sans-serif;}
target=_blank
}
div#terms{
        border-style:solid;
        position:relative;
        top:150px;
}
/*#terms {
   display:none;
   height:200px;
   overflow:auto;
   margin-bottom:1.5em;
   padding:10px;
   border:solid 1px #d7d7d7;
   color:#505050;
   background-color:#ffffff;
   font-size:90%;
}
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

4Store. (n.d.). *4Store Scarable RDF Storage*. Ανάκτηση από 4Store.org: http://4store.org/

Arenas, M., & Perez, J. (n.d.). Quering Semantic Web Data With Sparql. Chile.

Corno, F., & Farinetti, L. (n.d.). *Skos- Simple Knowledge Organization System*. Ανάκτηση από http://www.slideshare.net/fulvio.corno/5-Skos

DuCharm, B. (2013). *Learning Sparql.* America: O'Reilly Media.

Heath, T., & Bizer, C. (2011). *Linked Data: Evolving the Web into a Global Data Space.* Morgan & Claypool.

Herman, I. (2009, May 1). *Ivan's private site*. Ανάκτηση April 25, 2014, από http://ivan-herman.name/2009/05/01/library-of-congress-subject-headings-in-Skos-on-line/

W3C. (2014, February 25). *RDF 1.1 Turtle*. Ανάκτηση από Terse RDF Triple Language: http://www.w3.org/TR/2014/REC-turtle-20140225/

W3C. (2008, December 3). Cool URIs for the Semantic Web.

W3C. (2008, January 15). *SPARQL Query Language for RDF*. Ανάκτηση από http://www.w3.org/TR/rdf-sparql-query/

Wikipedia. (n.d.). *Multitier architecture*. Ανάκτηση από http://en.wikipedia.org/wiki/Multitier_architecture

Wikipedia. (n.d.). *Simple Knowledge Organization System*. Ανάκτηση από http://en.wikipedia.org/wiki/Simple_Knowledge_Organization_System

Ιόνιο Πανεπιστήμιο. (n.d.). *Internet Programming*. Ανάκτηση από di.ionio.gr: http://di.ionio.gr/~mistral/tp/intprogr/index.html

Καπιδάκης, Σ., Ελένη, Β., Κολοβιτσέα, Ό., & Κωνσταντοπούλου, Μ. (n.d.). Ψηφιακές Βιβλιοθήκες: Πόροι Μεταδεδομένων.

Παπαδάκης, Ι., Κυπριανός, Κ., Μαβροπόδη, Ρ., & Στεφανιδάκης, Μ. (2009, Σεπτέμβριος-Οκτώβριος). Subject-based Information Retrieval within Digital Libraries Employing LCSHs. *D-Lib Magazine*, σ. 9/10.

Linked Data - The Story So Far*International Journal on Semantic Web and Information Systems* 1-26

W3C *SKOS Simple Knowledge Organization System*

W3C Team Submission*Turtle - Terse RDF Triple Language*

W3C Working Group*SKOS*

W3C Working Group*SKOS Simple Knowledge Organization System Primer*