ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη Μοντέλου και Εφαρμογής Οπτικοποίησης Διασυνδεδεμένων Δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΜΕΛΙΝΑΣ ΣΚΟΥΡΛΑ

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Αθήνα, Δεκέμβριος 2013

Η σελίδα αυτή είναι σκόπιμα λευκή.

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

Ανάπτυξη Μοντέλου και Εφαρμογής Οπτικοποίησης Διασυνδεδεμένων Δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΜΕΛΙΝΑΣ ΣΚΟΥΡΛΑ

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 18η Δεκεμβρίου 2013.

.....	.....	.....
Ιωάννης Βασιλείου	Κώστας Κοντογιάννης	Νικόλαος Παπασπύρου
Καθηγητής Ε.Μ.Π.	Αναπληρωτής Καθηγητής Ε.Μ.Π.	Επίκουρος Καθηγητής Ε.Μ.Π.

Αθήνα, Δεκέμβριος 2013

.....

ΜΕΛΙΝΑ ΣΚΟΥΡΛΑ

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

Περίληψη

Ο ολοένα αυξανόμενος όγκος των διασυνδεδεμένων δεδομένων κάνει επιτακτική την ανάγκη εύρεσης τρόπων για την αποδοτική εξερεύνηση και την αξιοποίησή τους, ειδικά από χρήστες που δεν είναι εξοικειωμένοι με τις τεχνολογίες του σημασιολογικού ιστού. Στην εργασία μας αναπτύξαμε κατάλληλο μοντέλο οπτικής αναπαράστασης των Linked Datasets και μια web-based εφαρμογή οπτικοποίησης διασυνδεδεμένων δεδομένων που απευθύνεται και σε απλούς χρήστες μέσω ενός εύχρηστου User Interface που τους καθοδηγεί. Για το σκοπό αυτό σχεδιάστηκαν μηχανισμός συλλογής στατιστικών και κατάλληλο μοντέλο οπτικής αναπαράστασης των Linked Datasets, που είναι ανεξάρτητο από την εκάστοτε τεχνική και βιβλιοθήκη οπτικοποίησης και επιτρέπει τη διατήρηση στατιστικών στοιχείων για τα δεδομένα. Παράλληλα, για να γίνει δυνατή η οπτικοποίηση μεγάλου πλήθους δεδομένων, ομαδοποιεί τα δεδομένα και τα οργανώνει σε ιεραρχίες. Η εφαρμογή μπορεί να χρησιμοποιηθεί για οποιοδήποτε τύπο δεδομένων ανεξαρτήτως του τομέα στον οποίο ανήκουν και χειρίζεται αποδοτικά μεγάλα datasets, παρέχοντας δυνατότητα για overview των δεδομένων και zoom in στις περιοχές ενδιαφέροντος του χρήστη. Παρέχει τρεις διαφορετικές τεχνικές οπτικοποίησης (chart, timeline, treemap) σε συνδυασμό με παρουσίαση στατιστικών καθώς και πλοήγηση και επιλογή φίλτρων με γραφικό τρόπο (faceted search) στο set δεδομένων.

Λέξεις Κλειδιά: RDF οπτικοποίηση, Διασυνδεδεμένα δεδομένα, Μοντέλο οπτικής αναπαράστασης, Τεχνικές οπτικοποίησης, Διασυνδεδεμένα μεγάλα σύνολα δεδομένων, πλοήγηση και επιλογή φίλτρων με γραφικό τρόπο, overview, zoom in, επιλογή περιοχής ενδιαφέροντος, detail on demand

Η σελίδα αυτή είναι σκόπιμα λευκή.

Abstract

The constant increase of Linked Data on the web makes it imperative to find ways for their efficient exploration and exploitation, especially by users who are not familiar with the technologies of the Semantic Web. In our work, we developed a model for visual representation of Linked Datasets and a web-based application for visualizing linked data. Our web-application can be used by ordinary users through an intuitive User Interface that guides them. To that end, we designed a statistical data collection mechanism, and a model for visual representation of Linked Datasets, which is independent from specific techniques and visualization libraries and allows the retention of statistics for the data. At the same time, the application groups the data and organizes them in hierarchies. The application can be used for any data regardless of their type, and efficiently handles large datasets, providing data overview and allowing the users to zoom in on their areas of interest. It also provides three different visualization techniques (chart, timeline, treemap) combined with statistics presentation, and offers faceted filtering of the data set.

Keywords: RDF visualization, Linked Data, Visualization Model, Visualization Techniques, Linked Datasets, Large Datasets, Faceted Search, Overview, Zoom in, Data on Demand.

Η σελίδα αυτή είναι σκόπιμα λευκή.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Γιώργο Παπαστεφανάτο και τον κ. Νίκο Μπικάκη για τη βοήθειά τους στην εκπόνηση της παρούσας διπλωματικής εργασίας και τον επιβλέποντα καθηγητή μου κ. Ιωάννη Βασιλείου που μου έδωσε την ευκαιρία να ασχοληθώ με ένα πολύ ενδιαφέρον θέμα.

Ευχαριστώ ιδιαίτερα την οικογένειά μου για τη στήριξη που μου προσέφερε κατά τη διάρκεια των σπουδών μου και τους φίλους που ήταν πάντα δίπλα μου με τους χορούς, τις εκδρομές και την τρέλα τους κάνοντας την καθημερινότητα πιο όμορφη και δίνοντάς μου την ενέργεια για να συνεχίσω!

Η σελίδα αυτή είναι σκόπιμα λευκή.

Πίνακας περιεχομένων

1

Εισαγωγή

1.1 Οπτικοποίηση Linked Data – RDF δεδομένων

Τα τελευταία χρόνια παρατηρείται ολοένα και μεγαλύτερη αύξηση του όγκου σημασιολογικών δεδομένων στον Ιστό, με πλήθος φορέων να δημοσιεύουν τα δεδομένα τους ως διασυνδεδεμένα δεδομένα, και πρωτοβουλίες όπως τα Ανοιχτά Διασυνδεδεμένα Δεδομένα (Linked Open Data – LOD). Το τεράστιο αυτό πλήθος δεδομένων, ενώ ακολουθεί ένα κοινό πρότυπο κωδικοποίησης (RDF), δεν παύει να χαρακτηρίζεται από μεγάλη ετερογένεια όσον αφορά την ποιότητα των δεδομένων και τη σημασιολογία τους, καθιστώντας την αξιοποίησή τους ιδιαίτερα δύσκολη. Η οπτικοποίηση δεδομένων του σημασιολογικού ιστού και η παροχή φιλικών προς τον άνθρωπο μορφών αναζήτησης και πλοήγησης σε αυτά συμβάλλει σημαντικά στην ευρεία χρήση και αξιοποίησή τους από την πλειοψηφία των χρηστών του ιστού. Παρόλα αυτά, τα περισσότερα εργαλεία οπτικοποίησης απευθύνονται σε χρήστες που είναι γνώστες των τεχνολογιών του σημασιολογικού ιστού (1). Έτσι τα δεδομένα του σημασιολογικού Ιστού δεν είναι εύκολα προσβάσιμα και αξιοποιήσιμα από απλούς χρήστες. Παράλληλα, η εύρεση κατάλληλης οπτικοποίησης για ένα σύνολο δεδομένων δεν είναι απλή υπόθεση καθώς εξαρτάται από το είδος των δεδομένων που περιλαμβάνει (ιδιαίτερα ετερογενή συνήθως), τον όγκο τους και το σκοπό του χρήστη που τα εξερευνά. Εμπόδιο στην ευρύτερη χρήση διασυνδεδεμένων δεδομένων αποτελεί επίσης η δυσκολία να έχουμε μια εικόνα των διαθέσιμων datasets. Για την επαναχρησιμοποίηση, σύνδεση και εκτέλεση

ερωτημάτων σε ένα dataset είναι σημαντικό να γνωρίζουμε τη δομή και τα χαρακτηριστικά των δεδομένων του (1).

1.2 Αντικείμενο διπλωματικής

Στόχος της διπλωματικής είναι η ανάπτυξη μιας web-based εφαρμογής οπτικοποίησης και οπτικής πλοήγησης σε Linked datasets. Ιδιαίτερο βάρος δίνεται στην παροχή περισσότερων του ενός τρόπων οπτικοποίησης ενός set. Για το λόγο αυτό σχεδιάστηκε κατάλληλο μοντέλο οπτικής αναπαράστασης των Linked Data που είναι ανεξάρτητο από την εκάστοτε τεχνική και βιβλιοθήκη οπτικοποίησης και επιτρέπει τη διατήρηση στατιστικών στοιχείων για τα δεδομένα. Με βάση αυτό το μοντέλο δίνεται η δυνατότητα πλοήγησης και επιβολής φίλτρων με γραφικό τρόπο (faceted search) στο set δεδομένων.

Η μεγαλύτερη δυσκολία που καλείται να αντιμετωπίσει η εφαρμογή μας εντοπίζεται στο χειρισμό πολύ μεγάλου όγκου δεδομένων, καθώς το μέγεθος του Σημασιολογικού Ιστού αυξάνεται όλο και περισσότερο και η οπτική αναπαράσταση μπορεί να συμβάλει σημαντικά στην εξερεύνηση και αξιοποίησή του. Η εφαρμογή μας ομαδοποιεί τα δεδομένα και τα οργανώνει σε μια ιεραρχία, παρέχοντας αρχικά στο χρήστη μια γενική εποπτεία τους και δίνοντάς του τη δυνατότητα να πλοηγηθεί σε αυτά μέσω διαδραστικών διαγραμμάτων και να δει με περισσότερη λεπτομέρεια τις περιοχές που τον ενδιαφέρουν. Τα διαγράμματα παρέχουν επιπλέον πληροφορίες σχετικά με κάθε ομάδα δεδομένων μέσω tooltip που περιλαμβάνει στατιστικούς δείκτες και παραδείγματα δεδομένων που περιέχονται στην κάθε ομάδα. Ο χρήστης μπορεί να περιορίσει ακόμα περισσότερο τα δεδομένα στους τομείς που τον ενδιαφέρουν μέσω φίλτρων (facets). Για λόγους απόδοσης και ταχύτητας τα δεδομένα υφίστανται μια προεπεξεργασία για την κατασκευή του μοντέλου οπτικοποίησης και παρουσιάζονται μόνο εάν ζητηθούν (data on demand).

Για την καλύτερη κατανόηση της δομής και του περιεχομένου του dataset, παρουσιάζονται μαζί με την οπτικοποίηση και στατιστικοί δείκτες που περιγράφουν το dataset τόσο σε επίπεδο σχήματος όσο και σε επίπεδο δεδομένων. Επίσης ελέγχεται η ποιότητα των δεδομένων μέσω διαφόρων δεικτών που προκύπτουν από τα μεταδεδομένα (41). Για την επαναχρησιμοποίηση, σύνδεση και εκτέλεση ερωτημάτων σε ένα dataset είναι σημαντικό να γνωρίζουμε τη δομή και τα χαρακτηριστικά των δεδομένων του.

Άλλη μια πρόκληση που αντιμετωπίζει η εφαρμογή μας είναι η ετερογένεια που χαρακτηρίζει τα διασυνδεδεμένα δεδομένα και η επιλογή κατάλληλης οπτικοποίησης. Η εφαρμογή μας δεν περιορίζεται σε δεδομένα από συγκεκριμένους τομείς αλλά μπορεί να επεξεργαστεί οποιοδήποτε dataset χωρίς να απαιτεί τη γνώση του σχήματος και της σημασιολογίας ή του λεξιλογίου που το περιγράφει. Επιπλέον, λαμβάνοντας υπόψη μας ότι διαφορετικοί τύποι δεδομένων παρουσιάζονται καλύτερα με εξειδικευμένα για αυτούς διαγράμματα, παρέχουμε

ένα σύνολο από εναλλακτικούς τρόπους οπτικοποίησης. Συγκεκριμένα, χρησιμοποιούμε Treemap και Pie για παρουσίαση της ιεραρχίας κλάσεων, charts για αριθμητικά δεδομένα και timeline για ημερομηνίες.

Τέλος, ένα από τα μεγαλύτερα προβλήματα του Σημασιολογικού Ιστού αποτελεί η δυσκολία αξιοποίησης και εμπλουτισμού του από απλούς χρήστες. Τα περισσότερα εργαλεία οπτικοποίησης διασυνδεδεμένων – RDF δεδομένων απευθύνονται σε γνώστες των τεχνολογιών του σημασιολογικού ιστού, αποκλείοντας έτσι απλούς χρήστες που θα μπορούσαν να επωφεληθούν από αυτά. Η εφαρμογή μας υποστηρίζει εκτός από τους χρήστες-γνώστες και τους απλούς χρήστες, παρέχοντας ένα φιλικό User Interface που καθοδηγεί το χρήστη στην επιλογή ή φόρτωση δεδομένων, την επιλογή παραμέτρων οπτικοποίησης, την οπτικοποίηση και την πλοήγηση στο dataset. Δεν απαιτείται γνώση κάποιας γλώσσας ερωτημάτων (πχ. SPARQL) ή του τρόπου αναπαράστασης των δεδομένων.

1.2.1 Συνεισφορά

Η συνεισφορά της διπλωματικής συνοψίζεται ως εξής:

- Μελέτη και επιλογή τεχνικών οπτικής αναπαράστασης δεδομένων.
- Σχεδιασμός μοντέλου οπτικής αναπαράστασης μεγάλου όγκου RDF δεδομένων ανεξάρτητου από την εκάστοτε τεχνική και βιβλιοθήκη οπτικοποίησης που επιτρέπει την ομαδοποίηση και τη διατήρηση στατιστικών στοιχείων για τα δεδομένα.
- Προεπεξεργασία για την κατασκευή του μοντέλου και παρουσίαση δεδομένων μόνο όταν ζητηθούν (data on demand) με στόχο την απόδοση και ταχύτητα στη διαχείριση και πλοήγηση σε μεγάλα datasets.
- Συνδυασμός οπτικοποίησης και στατιστικών.
- Έλεγχος ποιότητας δεδομένων.

Ανάπτυξη Εφαρμογής

- Ανάπτυξη web-based εφαρμογής, προσβάσιμης ανεξαρτήτως πλατφόρμας.
- Γενικό εργαλείο, μπορεί να χρησιμοποιηθεί για οποιοδήποτε dataset ανεξαρτήτως του τομέα στον οποίο ανήκει, χωρίς να απαιτείται γνώση του σχήματος και της σημασιολογίας ή του λεξιλογίου που το περιγράφει.
- Οπτικοποίηση μεγάλων datasets με διαδραστικά διαγράμματα με δυνατότητα overview και zoom in στις περιοχές ενδιαφέροντος customizable από το χρήστη.
- Δυνατότητα πλοήγησης και επιβολής φίλτρων με γραφικό τρόπο (faceted search) στο set δεδομένων.
- Περισσότερα διαγράμματα από παρόμοια εργαλεία που περιγράφονται στη βιβλιογραφία.

- Υποστήριξη απλών χρηστών εκτός από γνώστες των τεχνολογιών σημασιολογικού Ιστού μέσω φιλικής διεπαφής (User Interface) που καθοδηγεί το χρήστη.

1.3 Οργάνωση κειμένου

Εργασίες σχετικές με το αντικείμενο της διπλωματικής παρουσιάζονται στο Κεφάλαιο 2. Το Κεφάλαιο 3 παρέχει το θεωρητικό υπόβαθρο για την κατανόηση της διπλωματικής. Στο Κεφάλαιο 4 περιγράφουμε τις απαιτήσεις και την αρχιτεκτονική του συστήματος. Το Κεφάλαιο 5 παρουσιάζει το σχεδιασμό του συστήματος και θέματα της διπλωματικής που έχουν τεχνικό ή αλγοριθμικό ενδιαφέρον. Στο Κεφάλαιο 6 ασχολούμαστε με τις λεπτομέρειες υλοποίησης. Στο Κεφάλαιο 7 παρουσιάζουμε αναλυτικά το σύστημα σύμφωνα με ένα σενάριο λειτουργίας. Το κεφάλαιο αυτό, λειτουργεί και ως εγχειρίδιο χρήσης του προγράμματος. Στο Κεφάλαιο 8 συνοψίζουμε τα αποτελέσματα της διπλωματικής και περιγράφουμε τα συμπεράσματα που προέκυψαν. Δίνουμε επίσης ιδέες για επέκταση της διπλωματικής. Στο Κεφάλαιο 9 δίνεται η βιβλιογραφία και οι πηγές που χρησιμοποιήθηκαν στο πλαίσιο της διπλωματικής.

2

Σχετικές εργασίες

Σχετικές εργασίες αφορούν εργαλεία που υποστηρίζουν οπτικοποίηση διασυνδεδεμένων δεδομένων, τεχνικές για οπτικοποίηση και διάδραση με μεγάλα datasets και υπολογισμό στατιστικών δεικτών σε διασυνδεδεμένα δεδομένα.

2.1 Οπτικοποίηση Διασυνδεδεμένων Δεδομένων

Αρκετές εργασίες αφορούν την εξερεύνηση (exploration) και οπτικοποίηση Διασυνδεδεμένων Δεδομένων. Οι Dadzie et al. (1) ανέλυσαν αρκετές από αυτές καταλήγοντας ότι στην πλειοψηφία τους έχουν σχεδιαστεί για tech-users και δεν παρέχουν καλή εποπτεία των διαθέσιμων δεδομένων.

Παρακάτω παρουσιάζονται οι βασικότερες κατηγορίες εργασιών στον τομέα της παρουσίασης και οπτικοποίησης Διασυνδεδεμένων Δεδομένων.

2.1.1 Desktop-based εφαρμογές

Οι υπάρχουσες desktop-based εφαρμογές απευθύνονται σε χρήστες που είναι γνώστες των τεχνολογιών του σημασιολογικού ιστού καθώς χρησιμεύουν στην ανάπτυξη OWL οντολογιών και RDF δεδομένων. Χαρακτηριστικά παραδείγματα είναι τα Protégé (2) , RDF Gravity (3) και Welkin (4). Τα εργαλεία αυτά διευκολύνουν την κατανόηση της δομής των δεδομένων παρέχοντας πολύπλοκες οπτικοποιήσεις με δέντρα και γράφους. Ο στόχος τους

όμως είναι η υποστήριξη ανάπτυξης οντολογιών και RDF δεδομένων και όχι η οπτικοποίηση διασυνδεδεμένων δεδομένων στον ιστό.

2.1.2 Παρουσίαση σε Πίνακες

Ένας από τους πιο συνηθισμένους τρόπους οπτικοποίησης RDF δεδομένων στο Web είναι η παρουσίαση των δεδομένων σε πίνακες. Η οπτικοποίηση των περισσότερων datasets διασυνδεδεμένων δεδομένων γίνεται με τον τρόπο αυτό πχ Freebase (5). Το Pubby (6) είναι ένα εργαλείο που οπτικοποιεί RDF δεδομένα σε μορφή πίνακα. Παρόλο που δίνει μια καλή εποπτεία των δεδομένων δεν παρέχει δυνατότητες κατηγοριοποίησης των δεδομένων ή εμπλουτισμό τους με επιπλέον σημασιολογία χρήσιμη για την παρουσίαση και κατανόησή τους από το χρήστη. Γενικότερα η οπτικοποίηση με πίνακες δεν προσφέρεται για ομαδοποίηση και παρουσίαση των δεδομένων με διαφορετικά χρώματα ούτε επιτρέπει στο χρήστη την αλληλεπίδραση με τα δεδομένα παρά μόνο μέσω του URI τους.

2.1.3 Browsers Διασυνδεδεμένων Δεδομένων

Οι Πλοηγοί (Browsers) διασυνδεδεμένων δεδομένων όπως οι Disco (7), Tabulator (8) και Explorator (9) επιτρέπουν στους χρήστες την πλοήγηση στα δεδομένα μέσα από γράφους και συνήθως παρουσιάζουν τα ζεύγη properties-values σε πίνακες. Δεν παρέχουν όμως μια γενική εποπτεία του dataset. Ο Rhizomer (10) παρέχει μια εποπτεία των datasets και επιτρέπει την αλληλεπίδραση με τα δεδομένα μέσω μενού πλοήγησης και facets, παρόλα αυτά δεν περιλαμβάνει οπτικοποίηση των δεδομένων.

2.1.4 Εργαλεία Οπτικοποίησης

Εργαλεία όπως τα Fenfire (11), RDF-Gravity (12) και IsaViz (13) παρέχουν οπτικοποίηση με γράφους. Παρόλο που οι γράφοι διευκολύνουν την κατανόηση της δομής του dataset, όταν το dataset είναι μεγάλο μπορεί ο γράφος να γίνει πολύ πολύπλοκος και δύσκολος στην κατανόηση και το χειρισμό του (14). Υπάρχουν επίσης εργαλεία που χρησιμοποιούν JavaScript για οπτικοποίηση RDF δεδομένων. Για παράδειγμα το Sgvizler (15) οπτικοποιεί τα αποτελέσματα SPARQL ερωτημάτων σε charts, maps, treemaps, κ.λπ. Όμως απαιτεί γνώση SPARQL, και δεν ανταποκρίνεται καλά στο χειρισμό και την παρουσίαση πολύ μεγάλων αποτελεσμάτων. Το LODWheel (16) παρέχει οπτικοποιήσεις με graphs, charts, maps για διάφορες κατηγορίες δεδομένων. Όμως εστιάζεται στην οπτικοποίηση vocabularies που χρησιμοποιούνται αυτήν τη στιγμή στη DBpedia. Το LODVisualization (17) παρέχει διαφορετικούς τρόπους οπτικοποίησης για οποιοδήποτε dataset. Δεν διαθέτει όμως timeline και chart για παρουσίαση αριθμητικών δεδομένων και ημερομηνιών.

2.1.5 Εργαλεία για συγκεκριμένους τομείς

Υπάρχουν αρκετά εργαλεία για οπτικοποίηση και πλοήγηση σε συγκεκριμένους τομείς. Για παράδειγμα τα LinkedGeoData browser (18) και map4rdf (19) αφορούν γεωχωρικά δεδομένα και ο FoaF Explorer χρησιμοποιείται για οπτικοποίηση RDF αρχείων που περιγράφονται με το vocabulary του FOAF.

2.1.6 Συνεισφορά του εργαλείου μας

Συνοψίζοντας, τα περισσότερα από τα υπάρχοντα εργαλεία δε διευκολύνουν τους απλούς χρήστες να κατανοήσουν και πλοηγηθούν σε διασυνδεδεμένα δεδομένα ή περιορίζονται σε συγκεκριμένους τομείς. Συχνά δεν παρέχουν εποπτεία των δεδομένων και της δομής τους ούτε παρουσιάζουν ιεραρχικά την πληροφορία με δυνατότητα zoom-in, παρά μόνο στην παρουσίαση ιεραρχίας κλάσεων ή properties. Έτσι αντιμετωπίζουν προβλήματα στο χειρισμό και την οπτικοποίηση μεγάλου όγκου δεδομένων.

Το εργαλείο μας από την άλλη μεριά παρέχει πολλούς τύπους διαγραμμάτων για διαφορετικά δεδομένα. Είναι ανεξάρτητο από συγκεκριμένο vocabulary, παρέχει γενική εποπτεία των δεδομένων και δίνει τη δυνατότητα πλοήγησης και παρουσίασης των λεπτομερειών. Αυτό του δίνει τη δυνατότητα για αποτελεσματικό χειρισμό μεγάλων datasets. Υποστηρίζει επίσης τους απλούς χρήστες εκτός από τους γνώστες του τομέα, μέσω ενός φιλικού user interface που καθοδηγεί το χρήστη για την επιλογή και φόρτωση δεδομένων και διευκολύνει την πλοήγηση και επιλογή των παραμέτρων οπτικοποίησης.

2.2 Οπτικοποίηση και διάδραση με μεγάλα datasets

Για την ανάλυση και οπτικοποίηση δεδομένων ο Shneiderman πρότεινε μια σειρά ενεργειών: "overview first, zoom and filter, then details on demand" (20).

Η εποπτεία είναι πολύ σημαντική για το χρήστη όταν πλοηγείται σε ένα dataset καθώς τον βοηθά να κατανοήσει τη συνολική δομή των δεδομένων ή μπορεί να χρησιμοποιηθεί σαν αφετηρία για την πλοήγηση. Το μεγάλο πλήθος και η ετερογένεια των δεδομένων όμως που χαρακτηρίζει τα διασυνδεδεμένα δεδομένα καθιστά δύσκολη την εποπτική τους παρουσίαση Μια συνηθισμένη λύση είναι να δομήσουμε τα datasets αυτά ιεραρχικά (21). Η ιεραρχική δομή επιτρέπει τη δημιουργία εποπτείας και επιπέδων που παρουσιάζουν τα δεδομένα με διαφορετικό βαθμό λεπτομέρειας. Υπάρχουν αρκετές τεχνικές για οπτικοποίηση ιεραρχιών. Για παράδειγμα, τα Treemaps (22) χρησιμοποιούν ορθογώνια για να δείξουν τη ρίζα και τα παιδιά της. Το μέγεθος του ορθογωνίου αντιπροσωπεύει το μέγεθος των δεδομένων που απεικονίζει (πλήθος, τιμή κλπ) συμπεριλαμβανομένων των παιδιών του. Στα CropCircles (23) κάθε κόμβος του δέντρου της ιεραρχίας αναπαρίσταται με ένα κύκλο. Τα παιδιά ενός κόμβου

βρίσκονται μέσα στον κύκλο του γονιού τους και η διάμετρος των κύκλων είναι ανάλογη με το μέγεθος των απογόνων τους. Τα SpaceTrees (24) μοιάζουν με τα συμβατικά δέντρα αλλά δίνουν επιπλέον τη δυνατότητα για zooming.

Το εργαλείο μας εφαρμόζει την αρχή του Shneiderman χρησιμοποιώντας Treemaps για απεικόνιση της ιεραρχίας κλάσεων. Επίσης για την απεικόνιση σε charts και timelines τα δεδομένα ομαδοποιούνται σε μια ιεραρχία με βάση την τιμή τους και ο χρήστης πλοηγείται από πάνω προς τα κάτω, από το μικρότερο επίπεδο λεπτομέρειας στο μεγαλύτερο.

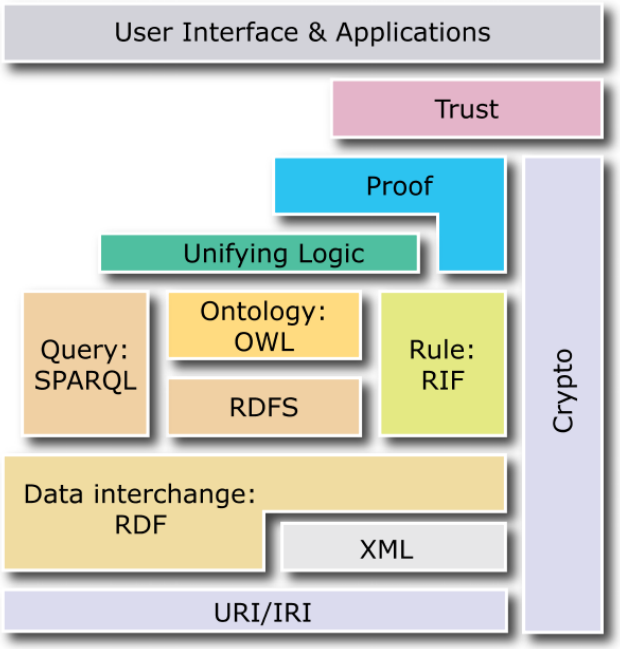
2.3 RDF Statistics

Οι εργασίες που έχουν γίνει στον τομέα των στατιστικών πάνω σε RDF δεδομένα είναι σχετικά περιορισμένες (25). Το make-void (26) είναι γραμμένο σε Java και χρησιμοποιεί το Jena toolkit (27) για να εισάγει RDF data και να παράγει στατιστικά που περιγράφονται με το το VoID (28) εκτελώντας SPARQL (29) ερωτήματα μέσω του SPARQL επεξεργαστή Jena ARQ (27). Το RDFStats (30) επίσης χρησιμοποιεί το παραπάνω προγραμματιστικό περιβάλλον και βιβλιοθήκες για επεξεργασία RDF δεδομένων αλλά χρησιμοποιεί τα στατιστικά που υπολογίζει για βελτιστοποίηση εκτέλεσης SPARQL ερωτημάτων. Το LODStats (25) βασίζεται στην ιδέα της επεξεργασίας μιας ροής από statements (statement-stream-based approach) για τον υπολογισμό στατιστικών από RDF datasets. Τα κύρια πλεονεκτήματά του σε σχέση με τα προηγούμενα εργαλεία είναι οι μικρότερες απαιτήσεις σε μνήμη και οι σημαντικά καλύτερες επιδόσεις του (25). Τα πλεονεκτήματα αυτά σε συνδυασμό με τη χρησιμότητα και το πλήθος των στατιστικών που υπολογίζει μας οδήγησαν στην χρήση μιας παραλλαγής του βασικού αλγορίθμου του για τον υπολογισμό στατιστικών στο εργαλείο μας.

3

Θεωρητικό υπόβαθρο

3.1 Βασικές έννοιες του Σημασιολογικού Ιστού



Σχήμα 1 Tim Berners-Lee 's Semantic Web Stack (2006)

3.1.1 Σημασιολογικός Ιστός

Ο σημασιολογικός ιστός είναι μια συλλογική κίνηση που προωθείται από το World Wide Web Consortium (W3C). Η βασική ιδέα πίσω από το σημασιολογικό ιστό είναι η ενσωμάτωση σημασιολογίας στα δεδομένα του ιστού και η αναπαράσταση της πληροφορίας με κοινά formats ώστε να είναι επεξεργάσιμη από μηχανές. Αυτή τη στιγμή το μεγαλύτερο μέρος της πληροφορίας του ιστού είναι προσβάσιμο μέσω ιστοσελίδων, δηλαδή HTML αρχείων που συνδέονται μεταξύ τους με υπερσυνδέσμους. Τα αρχεία αυτά μπορούν να διαβαστούν από ανθρώπους και μηχανήματα αλλά εκτός από την αναζήτηση με λέξεις-κλειδιά, τα μηχανήματα δεν μπορούν εύκολα να αποσπάσουν περισσότερες πληροφορίες από τα αρχεία καθεαυτά. Ο στόχος του σημασιολογικού ιστού είναι η συμπερίληψη σημασιολογικού περιεχομένου στις ιστοσελίδες, έτσι ώστε να περάσουμε από την τωρινή κατάσταση του ιστού (αδόμητα – ημιδομημένα αρχεία) σε έναν «Ιστό Δεδομένων». Με άλλα λόγια, θα είναι δυνατή η αναζήτηση, επεξεργασία, ανταλλαγή και επαναχρησιμοποίηση δεδομένων από μηχανές, σαν να είχαμε αποθηκευμένες τις πληροφορίες του ιστού σε μια τεράστια βάση δεδομένων. (31) (32)

3.1.2 Διασυνδεδεμένα Δεδομένα

Τα διασυνδεδεμένα δεδομένα (Linked Data) είναι μια βασική έννοια του σημασιολογικού ιστού. Αποτελούν μια μέθοδο δημοσίευσης δομημένων δεδομένων στον ιστό ώστε να μπορούν να συνδεθούν μεταξύ τους και με άλλα δεδομένα (interlinking). Η μέθοδος αυτή βασίζεται σε τεχνολογίες του ιστού όπως HTTP, RDF και URIs και στοχεύει στη δημοσίευση πληροφοριών σε μορφή αναγνώσιμη από υπολογιστές. Αυτό επιτρέπει τη διασύνδεση και αναζήτηση δεδομένων από διαφορετικές πηγές.

Οι βασικές αρχές των διασυνδεδεμένων δεδομένων συνοψίζονται ως εξής (33) (34):

- Χρήση URIs για αναπαράσταση αντικειμένων.
- Χρήση HTTP URIs για αναφορά και πρόσβαση στα αντικείμενα αυτά από ανθρώπους και μηχανήματα.
- Παροχή χρήσιμων πληροφοριών για ένα αντικείμενο από την ανάλυση του URI του με τη βοήθεια προτύπων όπως RDF, SPARQL.
- Παροχή συνδέσμων με άλλα σχετικά αντικείμενα μέσω του URI τους κατά τη δημοσίευση δεδομένων στον ιστό.

3.1.3 Ανοιχτά Διασυνδεδεμένα Δεδομένα

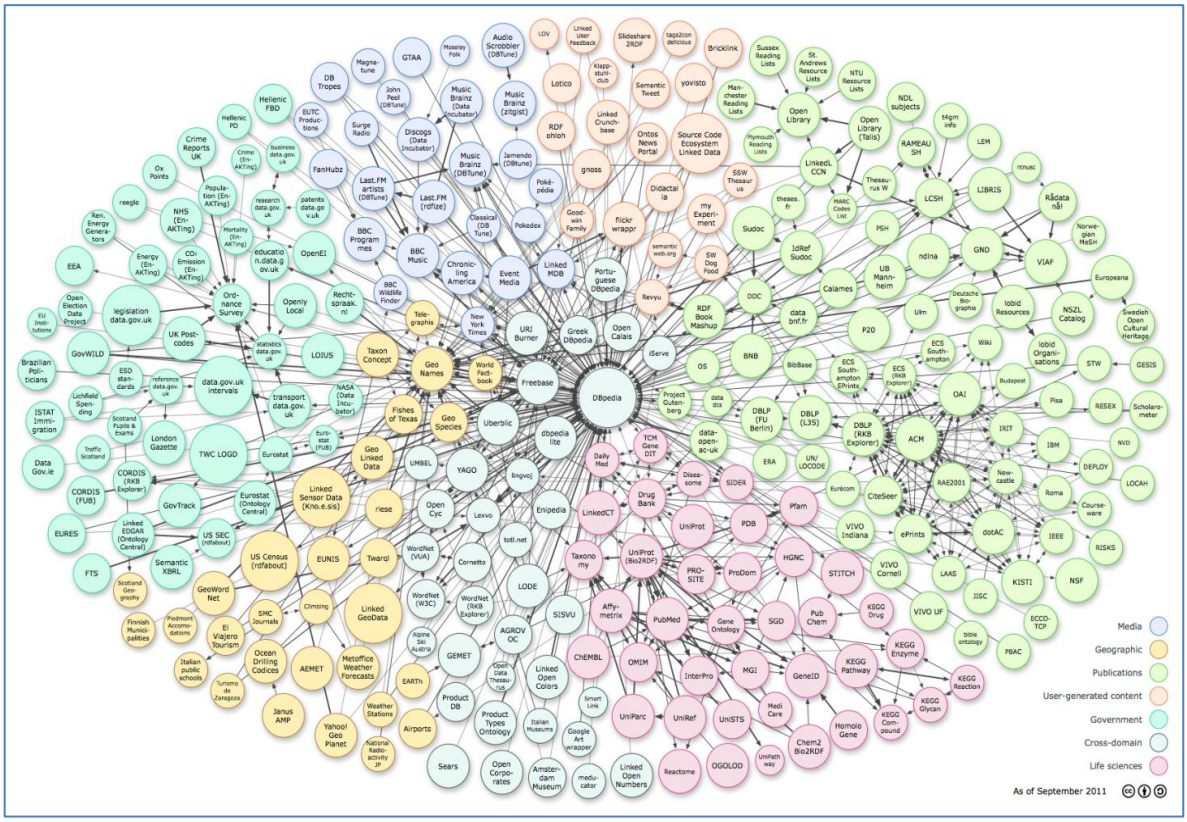
Τα ανοιχτά διασυνδεδεμένα δεδομένα – Linked Open Data (LOD) είναι μια κίνηση για τη δημοσίευση διαφόρων ανοιχτών datasets στον Ιστό σε RDF μορφή, φτιάχνοντας συνδέσμους

μεταξύ δεδομένων που προέρχονται από διαφορετικές πηγές. Όλο και περισσότεροι οργανισμοί και εταιρείες από διάφορους τομείς συμμετέχουν στην κίνηση αυτή δημοσιεύοντας τα δεδομένα τους στο σημασιολογικό Ιστό. Μερικά από τα πιο δημοφιλή datasets είναι το FOAF (περιγραφή ανθρώπων, των ιδιοτήτων και των σχέσεών τους), η DBpedia (δεδομένα από τη wikipeda), τα GeoNames (περιγραφή γεωγραφικών χαρακτηριστικών) κλπ. (35) (36)

Πίνακας 1 Δεδομένα ανά τομέα.

Domain	Number of datasets	Triples	%	(Out-)Links	%
Media	25	1,841,852,061	5.82 %	50,440,705	10.01 %
Geographic	31	6,145,532,484	19.43 %	35,812,328	7.11 %
Government	49	13,315,009,400	42.09 %	19,343,519	3.84 %
Publications	87	2,950,720,693	9.33 %	139,925,218	27.76 %
Cross-domain	41	4,184,635,715	13.23 %	63,183,065	12.54 %
Life sciences	41	3,036,336,004	9.60 %	191,844,090	38.06 %
User-generated content	20	134,127,413	0.42 %	3,449,143	0.68 %
	295	31,634,213,770		503,998,829	

(Στοιχεία από το State of the LOD Cloud document (http://lod-cloud.net/state/))



Σχήμα 2 Διάγραμμα του LOD Cloud. Κάθε κόμβος αντιπροσωπεύει ένα συγκεκριμένο dataset δημοσιευμένο σαν διασυνδεδεμένα δεδομένα. Οι ακμές υποδεικνύουν την ύπαρξη RDF συνδέσμων μεταξύ στοιχείων των δύο datasets που συνδέουν. Οι πιο έντονες ακμές αντιπροσωπεύουν το μεγάλο αριθμό συνδέσμων μεταξύ των δύο datasets ενώ οι αμφίδρομες ακμές τη συσχέτιση των datasets και από τις δύο πλευρές.

(Πηγή: http://lod-cloud.net/state/, http://lod-cloud.net/versions/2010-09-22/lod-cloud_colored.html)

3.1.4 RDF

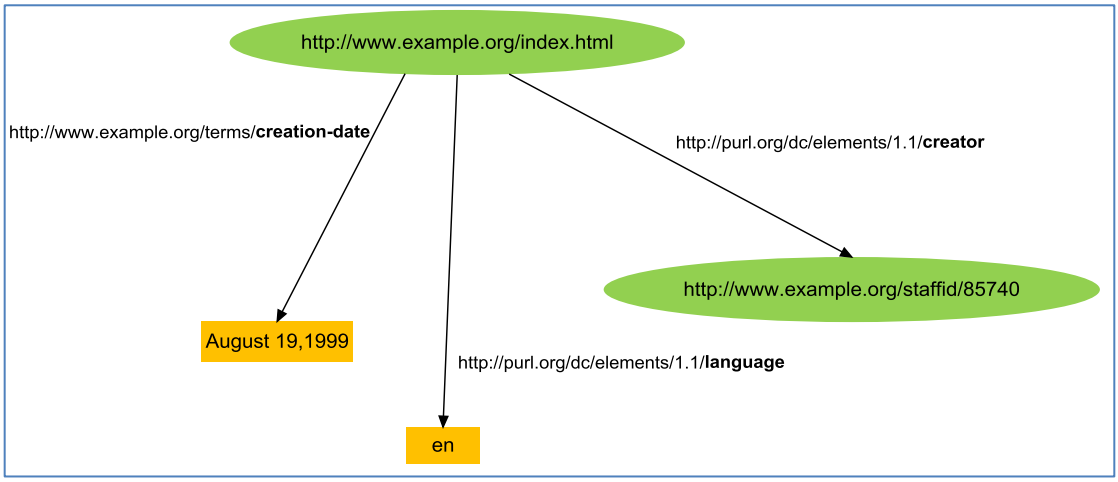
Το RDF (Resource Description Framework) είναι ένα πρότυπο – γλώσσα του W3C για την αναπαράσταση πόρων στον Ιστό και αποτελεί τη βάση του σημασιολογικού ιστού. Το RDF σχεδιάστηκε για να παρέχει ένα κοινό τρόπο περιγραφής πληροφοριών που να μπορεί να διαβαστεί και να κατανοηθεί από υπολογιστές.

Κάθε πόρος και κάθε ιδιότητα περιγράφονται από ένα μοναδικό IRI (Internationalized Resource Identifier) που τον χαρακτηρίζει. Η δομική μονάδα του προτύπου RDF είναι η πρόταση – statement. Μια πρόταση αποτελείται από το υποκείμενο (subject), το κατηγόρημα (predicate) και το αντικείμενο (object). Το υποκείμενο μπορεί να είναι πόρος (resource) που περιγράφεται από ένα IRI ή ανώνυμος – κενός κόμβος (blank node). Το κατηγόρημα αποτελεί την ιδιότητα που συνδέει το υποκείμενο με το αντικείμενο και περιγράφεται επίσης από ένα μοναδικό IRI. Το αντικείμενο αποτελεί την τιμή της ιδιότητας για το συγκεκριμένο υποκείμενο και μπορεί να είναι πόρος με συγκεκριμένο IRI, κενός κόμβος ή Literal. Τα Literals είναι τιμές διαφόρων τύπων δεδομένων όπως string, int, double, date κλπ και δεν περιγράφονται με IRI.

Σχήμα 3 RDF Statement

Πολλές RDF προτάσεις συνιστούν έναν RDF γράφο όπου οι κόμβοι αναπαριστούν τους πόρους (υποκείμενα, αντικείμενα) και οι ακμές τις μεταξύ τους σχέσεις (κατηγορήματα).

subject, predicate, object
http://www.example.org/index.html has a **creator** whose value is **John Smith**
http://www.example.org/index.html has **creation-date** the **August 16, 1999**
http://www.example.org/index.html has a **language** whose value is **English**



Σχήμα 4 Παράδειγμα αναπαράστασης RDF Statements σε RDF Graph

(Πηγή: http://www.w3.org/TR/rdf-primer/)

3.1.5 RDF Schema

Το RDF Schema (RDFS) είναι ένα πρότυπο του W3C και χρησιμοποιείται για να προσθέσει επιπλέον σημασιολογία σε RDF προτάσεις. Συγκεκριμένα ορίζει ένα λεξιλόγιο που επιτρέπει την περιγραφή ομάδων από συναφείς πόρους καθώς και των μεταξύ τους σχέσεων. Οι πόροι μπορούν να χωριστούν σε ομάδες που λέγονται κλάσεις. Οι πόροι που ανήκουν σε μια κλάση ονομάζονται στιγμιότυπα (instances) της κλάσης. Οι κλάσεις με τη σειρά τους μπορούν να οργανωθούν σε μια ιεραρχία (subclasses). Για την περιγραφή των ιδιοτήτων που χαρακτηρίζουν κλάσεις ή αντικείμενα/πόρους και την οργάνωσή τους σε μια ιεραρχία ορίζονται οι ιδιότητες (properties – subproperties) του RDFS. Το RDFS ορίζει επίσης τον τύπο/κλάση των πόρων που έχει νόημα να αποτελέσουν το υποκείμενο και αντικείμενο για μια δεδομένη ιδιότητα, όταν αυτή χρησιμοποιείται ως predicate σε μια πρόταση (domain και range αντίστοιχα). Για παράδειγμα, για την ιδιότητα income το υποκείμενο μπορεί να είναι τύπου Person ή Company και το αντικείμενο τύπου double. Το RDFS μπορεί επίσης να περιγράψει τον τύπο ενός αντικειμένου (resource, literal), τον τύπο (datatype) ενός literal (πχ string, integer κλπ), να παραθέσει σχόλια για έναν πόρο (comment) ή μια εκδοχή – όνομα του πόρου πιο αναγνώσιμη για τους ανθρώπους (label). Το λεξιλόγιο του RDFS, οι κλάσεις και οι ιδιότητες περιγράφονται από IRIs. Το λεξιλόγιο του RDFS αποτελείται από προκαθορισμένους πόρους των οποίων το URI έχει σαν πρόθεμα http://www.w3.org/2000/01/rdf-schema# (συναντάται και ως rdfs:). (37)

3.1.6 OWL

Η OWL (Web Ontology Language) είναι ένα πρότυπο του W3C που έχει ίδιο σκοπό με το RDFS και ουσιαστικά αποτελεί επέκτασή του με πλουσιότερη σημασιολογία. Επιτρέπει την έκφραση πιο πολύπλοκων σχέσεων μεταξύ πόρων δίνοντας έτσι τη δυνατότητα για εξαγωγή συμπερασμάτων (inference). Το λεξιλόγιο της OWL αποτελείται από προκαθορισμένους πόρους των οποίων το URI έχει σαν πρόθεμα http://www.w3.org/2002/07/owl# (συναντάται και ως owl:). Η OWL, για παράδειγμα, χρησιμοποιεί δικό της ορισμό κλάσεων (owl: Classes) και χωρίζει τις ιδιότητες σε αυτές που έχουν ως αντικείμενο πόρους (owl: ObjectProperty) και σε αυτές που έχουν κάποιο τύπο δεδομένων – literal (owl: DatatypeProperty). Οι ιδιότητες μπορούν να οριστούν σαν μεταβατικές (transitive), συμμετρικές (symmetric), αντίστροφες (inverse) κ.λπ. Η OWL μπορεί να παρέχει επίσης τελεστές σε σύνολα (owl:intersectionOf, owl:unionOf, owl:complementOf), περιορισμούς στις κλάσεις (owl:someValuesFrom, owl:hasValue, owl:cardinality), να ορίσει ισοδύναμες κλάσεις και ιδιότητες (owl:equivalentClass, owl:equivalentProperty), να δηλώσει ότι δύο αντικείμενα/πόροι είναι διαφορετικά (owl:sameAs, owl:differentFrom) κλπ. (38)

3.1.7 SPARQL

Η SPARQL είναι ένα πρότυπο του W3C και αποτελεί μια γλώσσα ερωτημάτων (query language) για RDF δεδομένα. Οι τύποι ερωτημάτων που υποστηρίζει είναι επιλογή (SELECT) δεδομένων με κάποια κριτήρια, η κατασκευή (CONSTRUCT) rdf γράφου, η ερώτηση (ASK) που εξετάζει αν ένα ερώτημα είναι δυνατό να γίνει πχ. αν υπάρχει μια συγκεκριμένη δομή στα δεδομένα, και η περιγραφή (DESCRIBE) που μπορεί να περιγράψει τη δομή του RDF γράφου που εξετάζουμε. Η αποτίμηση των ερωτημάτων της SPARQL βασίζεται στο ταίριασμα προτύπων σε γράφο (Graph Pattern Matching). Ένα πρότυπο τριπλέτας (triple pattern) αποτελείται από μια RDF triple και μεταβλητές που μπορούν να αντιπροσωπεύουν τα υποκείμενο, κατηγόρημα, αντικείμενο. Ένα Βασικό Πρότυπο Γράφου (Basic Graph Pattern) είναι μια σειρά από triple patterns και περιορισμούς (filters). Ένα Πρότυπο Γράφου (Graph Pattern) αποτελείται από βασικά πρότυπα γράφου μαζί με τελεστές σύζευξης (AND), ένωσης (UNION), επιλεκτικής συμπερίληψης προτύπων (OPTIONAL) και περιορισμών (FILTER).

Η δομή SPARQL ερωτημάτων είναι η παρακάτω:

- Μια συνθήκη SELECT που ορίζει τις μεταβλητές που θα εμφανιστούν στα αποτελέσματα του ερωτήματος. Τα αποτελέσματα μπορούν επίσης να επιστραφούν σε μορφή γράφου (DESCRIBE, CONSTRUCT) ή ως μια απάντηση TRUE/FALSE (ASK).
- Μια συνθήκη FROM που ορίζει τους γράφους στους οποίους θα εκτελεστεί το ερώτημα.
- Μια συνθήκη WHERE που παρέχει το Βασικό Πρότυπο Γράφου (Basic Graph Pattern) το οποίο θα ταιριάξει με τα δεδομένα του γράφου που «ρωτάμε».
- Πρόσθετες επιλογές που αφορούν την ομαδοποίηση των δεδομένων (GROUP BY) ή την παρουσίασή τους (ORDER BY, LIMIT κλπ).

Data:

```
# Default graph (stored at http://example.org/foaf/aliceFoaf)
@prefix  foaf:  <http://xmlns.com/foaf/0.1/> .
```

Query:

```
# Default graph (stored at http://example.org/foaf/aliceFoaf)
@prefix  foaf:  <http://xmlns.com/foaf/0.1/> .

_:a  foaf:name     "Alice" .
_:a  foaf:mbox     <mailto:alice@work.example> .

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT  ?name
FROM    <http://example.org/foaf/aliceFoaf>
WHERE   { ?x foaf:name ?name }
```

Result:

name
"Alice"

Σχήμα 5 Παράδειγμα SPARQL ερωτήματος. Πηγή: (39)

3.1.8 Named Graphs

Ένας Named Graph είναι ένας RDF γράφος (ένα σύνολο από RDF Statements) που χαρακτηρίζεται από ένα δικό του URI. Με τον τρόπο αυτό οι RDF Γράφοι μπορούν να ξεχωρίζουν μεταξύ τους όταν δημοσιεύονται στον Ιστό. Παρότι οι Named Graphs μοιάζουν με απλά δισυνδεδεμένα αρχεία (διασυνδεδεμένα δεδομένα) όταν δημοσιεύονται στον Ιστό, αποδεικνύονται ιδιαίτερα χρήσιμοι στο χειρισμό συνόλων RDF δεδομένων μέσα σε ένα RDF Store. Συγκεκριμένα, μπορούμε να περιορίσουμε την εμβέλεια ενός SPARQL ερωτήματος σε ένα συγκεκριμένο σύνολο από named graphs. Οι named graphs αναπαρίστανται με τετράδες (quads). Κάθε statement – triple έχει ένα επιπλέον πεδίο που υποδηλώνει το γράφο στον οποίο ανήκει το statement:

<*graphname*> <*subject*> <*predicate*> <*object*>

Το <*graphname*> είναι και αυτό ένα URI.

Η ιδέα των quads προέκυψε από την ανάγκη ένταξης RDF Statements σε διαφορετικά contexts:

<*context*> <*subject*> <*predicate*> <*object*>

Έτσι στην εφαρμογή μας χρησιμοποιούμε την τεχνική των Named Graphs για να χωρίσουμε τα δεδομένα μας σε διαφορετικές κατηγορίες. (40)

3.2 Βασικές αρχές οπτικοποίησης με έμφαση στα Linked Data

3.2.1 Απαιτήσεις για user interfaces εργαλείων οπτικοποίησης

Ο Shneiderman ορίζει στο mantra του σχετικά με την εύρεση οπτικών πληροφοριών (visual-information-seeking mantra) (20) τις απαιτήσεις υψηλού επιπέδου για διεπαφές χρήστη εργαλείων οπτικοποίησης:

- Δυνατότητα εποπτικής παρουσίασης των δεδομένων.
- Υποστήριξη επικέντρωσης στις επιλεγμένες περιοχές ενδιαφέροντος αγνοώντας τα λιγότερο σημαντικά δεδομένα.
- Υποστήριξη οπτικοποίησης λεπτομερειών στις περιοχές ενδιαφέροντος.

3.2.2 Οδηγίες σχεδιασμού εργαλείων οπτικοποίησης και ανάλυσης για Linked Data

3.2.2.1 Οδηγίες σχεδιασμού - Οπτική αναπαράσταση και ανάλυση

- Οπτική αναπαράσταση: Η χρήση συχνά διαδραστικών οπτικών αναπαραστάσεων, γραφικών, εικόνων κ.λπ. που αξιοποιούν την ανθρώπινη αντίληψη για να βοηθήσουν στην κατανόηση πολύπλοκων δομών δεδομένων και του περιεχομένου τους με στόχο την καλύτερη αναζήτηση, παρουσίαση και ανάλυση των δεδομένων.
- Εποπτεία (overview) δεδομένων: Συνολική άποψη των δεδομένων, χρήσιμη στην παρουσίαση της συνολικής δομής των δεδομένων.
- Λεπτομέρειες κατ' απαίτηση: Σε συνδυασμό με το overview των δεδομένων επιτρέπει στο χρήστη να επικεντρωθεί στη λεπτομέρεια των περιοχών που τον ενδιαφέρουν ώστε να αναλύσει καλύτερα τα δεδομένα που διαθέτει.
- Επισήμανση συνδέσμων στα δεδομένα: Η αναγνώριση των τύπων και της ισχύος των συνδέσμων ή των σχέσεων μέσα σε ένα dataset ή μεταξύ διαφορετικών datasets συνεισφέρει σημαντικά στην κατανόηση των δεδομένων και την ανακάλυψη γνώσης.
- Υποστήριξη επεκτασιμότητας (scalability): Χρειάζεται η διαχείριση μεγάλου πλήθους από πολύπλοκα και ετερογενή διασυνδεδεμένα δεδομένα, συχνά αποθηκευμένα σε απομακρυσμένες τοποθεσίες.
- Δυνατότητα επερωτήσεων: Είναι ιδιαίτερα χρήσιμη η δυνατότητα εκτέλεσης ερωτημάτων με μια γλώσσα ερωτημάτων όπως η SPARQL, που απευθύνεται σε χρήστες με την απαραίτητη τεχνογνωσία, εκτός από μεθόδους που εξυπηρετούν τους απλούς χρήστες (πχ. αναζήτηση με λέξεις-κλειδιά, User Interfaces με φόρμες ερωταπαντήσεων και μενού πλοήγησης κλπ.)
- Φιλτράρισμα: Το φιλτράρισμα επιτρέπει την επισήμανση των περιοχών ενδιαφέροντος διώχνοντας από το προσκήνιο τις λιγότερο σχετικές πληροφορίες.
- Διατήρηση Ιστορικού: Η διατήρηση ιστορικού βοηθά στην επανεξέταση ή την επανεκτέλεση ενεργειών, στην επιστροφή σε συγκεκριμένα σημεία της πλοήγησης, στην αναίρεση ενεργειών κλπ.

  Πηγή: (1)

3.2.2.2 Οδηγίες σχεδιασμού – Χρήση Διασυνδεδεμένων Δεδομένων

- Πρότυπα παρουσίασης (presentation templates): Προκαθορισμένες συχνά δομές που αντιστοιχίζουν κάποια χαρακτηριστικά των δεδομένων σε συγκεκριμένες οπτικές

αναπαραστάσεις πχ. ένας foaf:Person αναπαρίσταται με το ονοματεπώνυμο και τη φωτογραφία του.

- Σημείο εισόδου: Ο προκαθορισμένος τρόπος εισόδου στο LOD είναι μέσω ενός URI. Για να διευκολυνθεί η πρόσβαση και από απλούς χρήστες που δεν είναι γνώστες του τομέα είναι χρήσιμη η αναζήτηση με λέξεις-κλειδιά και άλλες τεχνικές που καθοδηγούν το χρήστη πχ κλικ σε ένα διάγραμμα για την επιλογή ενός πόρου κ.λπ.
- Ανεξαρτησία από τον τομέα: Λόγω του πλήθους και της ετερογένειας των διασυνδεδεμένων δεδομένων είναι ιδιαίτερα χρήσιμη η ανάπτυξη εργαλείων αναζήτησης και επεξεργασίας ανεξάρτητων από ένα τομέα, ώστε να επιτρέπουν την πλοήγηση και το χειρισμό του συνόλου των διαθέσιμων δεδομένων.
- Πλοήγηση / αναζήτηση με facets: Η χρήση facets συμβάλλει στην καλύτερη αναζήτηση και πλοήγηση σε μεγάλο πλήθος δεδομένων, βοηθώντας το χρήστη να δει τα χαρακτηριστικά τους και τις μεταξύ τους σχέσεις και να επιλέξει τις περιοχές που τον ενδιαφέρουν.
- Δημοσίευση: Οι απλοί χρήστες αλλά και οι γνώστες των τεχνολογιών του σημασιολογικού ιστού χρειάζονται υποστήριξη στην κωδικοποίηση και δημοσίευση νέων δεδομένων στο LOD. Επιπλέον, οι τελικοί χρήστες χρειάζονται μεθόδους επισήμανσης σφαλμάτων και επικύρωσης νέων δεδομένων και συνδέσμων με υπάρχοντα διασυνδεδεμένα δεδομένα.
- Συγχώνευση δεδομένων: Τα διασυνδεδεμένα δεδομένα περιλαμβάνουν ετερογενή δεδομένα από διάφορες πηγές. Για την αποδοτικότερη διαχείριση των διασυνδεδεμένων δεδομένων είναι καλή πρακτική να συνδέονται τα δεδομένα που συσχετίζονται και να επαναχρησιμοποιούνται τα χαρακτηριστικά που είναι κοινά στα δεδομένα.
- Εγκυρότητα / προέλευση δεδομένων: Η επικύρωση της προέλευσης των δεδομένων επηρεάζει την εμπιστοσύνη όσον αφορά στην αξιοπιστία και την ποιότητά τους.
- Επεξεργασία δεδομένων: Είναι χρήσιμο να έχουν οι χρήστες τη δυνατότητα να επέμβουν στα δεδομένα εμπλουτίζοντάς τα και διορθώνοντας σφάλματα.
- Επαναχρησιμοποιήσιμη κωδικοποίηση αποτελεσμάτων: Η κωδικοποίηση αποτελεσμάτων με χρήση πρότυπων οντολογιών και λεξιλογίων εξασφαλίζει τη σωστή ερμηνεία, την επαναχρησιμοποίηση και τον εμπλουτισμό τους καθώς και τη διασύνδεσή τους με άλλα δεδομένα.

  Πηγή: (1)

3.3 Στατιστική Ανάλυση

Για τον υπολογισμό στατιστικών στα δεδομένα μας εφαρμόζουμε τον αλγόριθμο που περιγράφεται στο (25).

Ο αλγόριθμος (Αλγόριθμος 1) βασίζεται στην εξής ιδέα: Ορίζουμε κάποια κριτήρια και στη συνέχεια διαβάζουμε μια ροή RDF προτάσεων (από RDF αρχείο, SPARQL endpoint). Για κάθε κριτήριο που ικανοποιεί κάθε πρόταση εκτελούμε κάποιες προκαθορισμένες ενέργειες, Μέσω αυτών των ενεργειών ή της επεξεργασίας τους μετά το τέλος του αλγορίθμου προκύπτουν τα ζητούμενα στατιστικά. Με τον τρόπο αυτό τα δεδομένα διατρέχονται μόνο μία φορά, γεγονός που συμβάλλει στην πολύ καλή απόδοση του αλγορίθμου.

3.3.1 Ορισμός Στατιστικών Κριτηρίων

Παρακάτω παρουσιάζουμε τον τυπικό ορισμό των στατιστικών κριτηρίων.

Ορισμός 1 (Στατιστικά κριτήρια). Ένα στατιστικό κριτήριο είναι μια τριάδα (F, D, P), όπου:

- **F μια SPARQL συνθήκη φιλτραρίσματος.**
- **D μια δομή δεδομένων για την αποθήκευση ενδιάμεσων αποτελεσμάτων και περιγραφή του τρόπου εισαγωγής δεδομένων από τη ροή RDF statements στη δομή αυτή μετά την επιβολή της F.**
- **P ένα φίλτρο που επιβάλλεται μετά την επεξεργασία στη δομή δεδομένων D.**

Το F καθορίζει αν η τριπλέτα – πρόταση που επεξεργαζόμαστε πρέπει να αλλάξει κάτι σε μια δομή δεδομένων D. Για κάθε τριπλέτα εξετάζονται όλα τα κριτήρια. Παραδείγματα κριτηρίων είναι: αν το αντικείμενο της τριπλέτας είναι literal, αν το κατηγόρημα ανήκει σε κάποιο δεδομένο λεξιλόγιο κλπ.

Οι δομές δεδομένων D που χρησιμοποιούνται περιλαμβάνουν τα HashMap (M), Set (S), directed Graph (G). Παράδειγμα κριτηρίου είναι M[?predicate]++. Το ?predicate είναι μια μεταβλητή που αντιπροσωπεύει στοιχείο της τριπλέτας που εξετάζουμε (εδώ το κατηγόρημα). Ο συμβολισμός που χρησιμοποιούμε υποδηλώνει ότι αυξάνεται κατά ένα η τιμή (value) του κλειδιού (key) του HashMap M με τιμή ?predicate. Με το συγκεκριμένο κριτήριο μετράμε το πλήθος του κάθε property.

Στην πιο απλή περίπτωση το P επιστρέφει τις τιμές μιας δομής δεδομένων D. Άλλες φορές χρησιμοποιείται για να επιλέξει τα top-k στοιχεία μιας δομής D ή να εκτελέσει κάποιους επιπλέον υπολογισμούς πχ. να υπολογίσει το πλήθος ή το μέσο όρο των στοιχείων της δομής.

Algorithm 1: Evaluation of a set of statistical criteria on a triple stream.

```
Data: CriteriaSet CS; TripleInputStream S
forall the Criteria C ∈ CS do
    initialise datastructures
while S.hasNext() do
    Triple T = S.next();
    forall the Criteria C ∈ CS do
        if T satisfies C.T and C.F then
            execute C.D;
            if datastructures exceed threshold then
                purge datastructures;
forall the Criteria C ∈ CS do
    execute C.P and return results
```

Auer, Sören, et al., et al. (2012) LODStats - An Extensible Framework for High-performance Dataset Analytics. *Proceedings of the 18th international conference on Knowledge Engineering and Knowledge Management*, *pp*. 353-362.

4

Περιγραφή Συστήματος

Ακολουθεί η περιγραφή της αρχιτεκτονικής του συστήματος και η ανάλυση απαιτήσεων για τις λειτουργίες του.

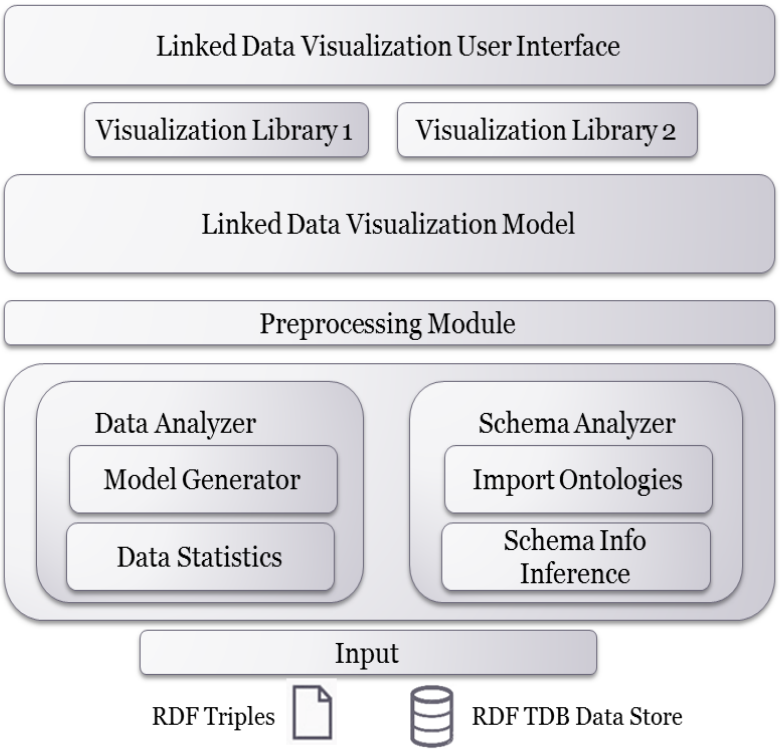
4.1 Αρχιτεκτονική

Το σύστημά μας αποτελείται από τα εξής υποσυστήματα (Σχήμα 6):

1. Input: Εισαγωγή δεδομένων από αρχείο ή URI αρχείου με RDF Triples ή από TDB Data Store.
2. Preprocessing: Προεπεξεργασία δεδομένων.
    - 2.1 Data analyzer
        - 2.1.1 Model Generator: Παραγωγή μοντέλου για την οπτικοποίηση δεδομένων (βλ. και 2.3). Πιο συγκεκριμένα, γίνεται υπολογισμός των απαραίτητων τιμών και ομαδοποίηση δεδομένων.
        - 2.1.2 Data Statistics: Υπολογισμός στατιστικών για το dataset.
    - 2.2 Schema analyzer
        - 2.2.1 Import Ontologies: Εντοπισμός vocabularies και import των αντίστοιχων οντολογιών.

2.2.2 Schema Info Inference: Εύρεση της ιεραρχίας κλάσεων, αριθμού instances κάθε κλάσης, domain και range των numeric και date properties.

3. Linked Data Visualization Model: Το τελικό μοντέλο που παράγεται από τις λειτουργίες των παραπάνω συστημάτων λαμβάνοντας υπόψη τις επιλογές του χρήστη και αποτελεί τη βάση για την οπτικοποίηση.
4. Visualization Libraries: Βιβλιοθήκες που χρησιμοποιούνται στην οπτικοποίηση του μοντέλου με διαφορετικούς τρόπους.
5. Linked Data Visualization User Interface: Διεπαφή χρήστη που επιτρέπει επιλογή/φόρτωση δεδομένων, επιλογή στοιχείων του dataset που θα οπτικοποιηθούν, οπτικοποίηση, πλοήγηση στο dataset και στα διαγράμματα.



Σχήμα 6 Αρχιτεκτονική του Συστήματος

4.2 Περιγραφή Λειτουργιών

4.2.1 Input

Το υποσύστημα αυτό είναι υπεύθυνο για την εισαγωγή δεδομένων και προαιρετικά οντολογίας στο σύστημα. Η εισαγωγή δεδομένων μπορεί αν γίνει με δύο τρόπους:

- Εισαγωγή δεδομένων ήδη αποθηκευμένων σε ένα TDB Data Store.
- Εισαγωγή δεδομένων και προαιρετικά οντολογίας από RDF αρχείο. Τα αρχεία αυτά βρίσκονται στο τοπικό σύστημα του χρήστη, ο οποίος μπορεί να τα ανεβάσει στο σύστημά μας μέσω του User Interface.
- Εισαγωγή δεδομένων και προαιρετικά οντολογίας από URI online RDF αρχείου το οποίο μπορεί να εισάγει ο χρήστης στο σύστημά μας μέσω του User Interface.

Ο χρήστης επιλέγει μέσω του UI τον τρόπο εισαγωγής των δεδομένων ανεβάζοντας τα αρχεία του, δίνοντας το URI τους ή επιλέγοντας από μια λίστα ήδη φορτωμένων datasets.

Η εισαγωγή δεδομένων είναι απαραίτητη για τη λειτουργία των υπόλοιπων υποσυστημάτων.

4.2.2 Preprocessing

Το υποσύστημα αυτό εκτελεί την προεπεξεργασία των δεδομένων. Καλεί τα υποσυστήματα που είναι υπεύθυνα για κάθε τμήμα της προεπεξεργασίας με τις κατάλληλες παραμέτρους.

- Αν η προεπεξεργασία αφορά δεδομένα που έχει εισάγει ο χρήστης από δικά του αρχεία ή URI, το μοντέλο που προκύπτει από την προεπεξεργασία αποθηκεύεται στη μνήμη.
- Διαφορετικά ελέγχεται αν το μοντέλο της προεπεξεργασίας είναι ήδη αποθηκευμένο στο TDB Store. Αν ναι, γίνεται σύνδεση με αυτό. Αν όχι, εκτελείται η προεπεξεργασία και το μοντέλο που προκύπτει αποθηκεύεται στο TDB Store.

4.2.2.1 Schema Analyzer

4.2.2.1.1 Import Ontologies

Εισάγει τις οντολογίες που εντοπίζονται στο dataset.

4.2.2.1.2 Schema Info Inference

Με χρήση μηχανισμών εξαγωγής συμπερασμάτων (inference)

- Υπολογίζεται η ιεραρχία κλάσεων.
- Υπολογίζεται ο αριθμός instances κάθε κλάσης.

- Για κάθε κλάση υπολογίζονται τα datatype και object properties που συνδέονται με αυτή, μαζί με τον αριθμό εμφανίσεών τους.
- Για κάθε numeric και date property υπολογίζονται οι κλάσεις που αποτελούν το domain και το range του, και ο αριθμός εμφανίσεών του.

4.2.2.2 Data analyzer

4.2.2.2.1 Data Statistics

Υπολογίζει στατιστικά για το dataset και συλλέγει πληροφορίες από μετα-δεδομένα διατρέχοντας όλο το dataset πριν την εισαγωγή οντολογιών.

4.2.2.2.2 Model Generator

- Επιλέγει τα δεδομένα που θα συμπεριληφθούν στο μοντέλο. Τα δεδομένα αυτά είναι τριπλέτες που έχουν ως object αριθμητική τιμή ή ημερομηνία.
- Χωρίζει τις επιλεγμένες τριπλέτες σε ομάδες με βάση το predicate και την κλάση στην οποία ανήκει το subject.

4.2.3 Linked Data Visualization Model

Διαμορφώνεται το τελικό μοντέλο με αξιοποίηση των αποτελεσμάτων της προεπεξεργασίας. Με δεδομένα συγκεκριμένα properties (numeric και date properties) και προαιρετικά κλάσεις, επιλέγει τις τριπλέτες με predicate από τα δεδομένα properties και subject που ανήκει σε κάποια από τις κλάσεις που δίνονται. Χωρίζει τις επιλεγμένες τριπλέτες σε ομάδες με βάση το predicate και την τιμή του object, τις οποίες στη συνέχεια οργανώνει σε μια ιεραρχία. Το βάθος της ιεραρχίας, ο αριθμός των στοιχείων κάθε ομάδας και το εύρος της υπολογίζεται από παραμέτρους που δίνονται στο σύστημα.

Το μοντέλο αυτό παρέχει τα δεδομένα για τη δημιουργία και πλοήγηση σε διαφορετικές οπτικοποιήσεις και για την αλληλεπίδραση του χρήστη με τα δεδομένα μέσω του UI.

4.2.4 Visualization Libraries

Διατηρούμε δύο διαφορετικές βιβλιοθήκες για δημιουργία διαφορετικών οπτικοποιήσεων. Η μία βιβλιοθήκη (Highcharts) σχεδιάζει Charts και Timelines ενώ η δεύτερη (Google Charts) Treemaps και Pie.

- Κάθε βιβλιοθήκη δέχεται σαν είσοδο δεδομένα από το μοντέλο, τα οποία οπτικοποιεί.

- Tooltip on mouseover σε στοιχεία των διαγραμμάτων για παροχή επιπλέον πληροφοριών στο χρήστη.
- Εκτέλεση ενεργειών όταν ο χρήστης κάνει κλικ σε στοιχείο (πχ. bar) του διαγράμματος.

4.2.5 Linked Data Visualization User Interface

Η διεπαφή χρήστη που επιτρέπει την αλληλεπίδρασή του με το σύστημα. Συγκεκριμένα παρέχει τις εξής λειτουργίες:

- Δυνατότητα επιλογής ενός ήδη φορτωμένου στο σύστημα dataset (από το TDB Store) μέσω ενός μενού επιλογών.
- Φόρτωση τοπικών RDF αρχείων δεδομένων (και οντολογίας) του χρήστη. Παρέχονται τέσσερις επιλογές για την κωδικοποίηση των αρχείων αυτών (RDF/XML, N-TRIPLE, TURTLE, N3).
- Φόρτωση RDF αρχείων δεδομένων (και οντολογίας) από το χρήστη δίνοντας το URI τους.
- Παρουσίαση στο χρήστη στατιστικών δεικτών και πληροφοριών σχετικά με το dataset που επέλεξε/ανέβασε.
- Εμφάνιση τριών facets μετά τη φόρτωση των δεδομένων. Το 1^{ο} παρουσιάζει τις κλάσεις του dataset σε μια δενδρική δομή. Αρχικά παρουσιάζονται οι κλάσεις που βρίσκονται στην κορυφή της ιεραρχίας. Ο χρήστης μπορεί με κλικ σε μία κλάση να δει τις υποκλάσεις της και να πλοηγηθεί έτσι στην ιεραρχία κλάσεων. Το 2^{ο} facet παρουσιάζει τα properties που έχουν ως range αριθμητικά δεδομένα του dataset, ενώ το 3^{ο} τα properties με range ημερομηνίες. Τόσο οι κλάσεις όσο και τα properties παρουσιάζονται αρχικά ταξινομημένα σε φθίνουσα σειρά με βάση τον αριθμό των εμφανίσεών τους στο dataset, με δυνατότητα ταξινόμησης σε αλφαβητική σειρά. Τα στοιχεία των facets έχουν ένα checkbox το καθένα, που επιτρέπει στο χρήστη την επιλογή αυτών που θέλει να οπτικοποιήσει. Σε κάθε στοιχείο των facets εμφανίζεται tooltip με το πλήρες URI του στοιχείου και επιπλέον στα properties οι κλάσεις που έχουν ως domain και range. Επιλέγοντας ένα στοιχείο από ένα facet (κάνοντας κλικ στο checkbox στα αριστερά του) τα υπόλοιπα facets φιλτράρονται αναλόγως. Συγκεκριμένα, όταν επιλέγεται μια κλάση, στα facets των numeric και date properties μένουν ορατά μόνο τα properties που έχουν την κλάση αυτή ως domain. Αντιστρόφως, όταν επιλεγεί ένα property (είτε numeric είτε date), στο facet των κλάσεων μένουν ορατές μόνο οι κλάσεις που έχει ως domain. Τα numeric και date facets δε σχετίζονται μεταξύ τους.

- Οπτικοποίηση των στοιχείων που επελέγησαν στα facets, σε διαφορετικά interactive διαγράμματα με παροχή στατιστικών και δυνατότητα προσθήκης και διαγραφής facets. Ο χρήστης μπορεί να επιλέξει μεταξύ chart, timeline, treemap.
    - **Treemap:** Αρχικά παρουσιάζονται οι ανώτερες στην ιεραρχία κλάσεις που επέλεξε ο χρήστης. Τα δεδομένα κάθε κλάσης φιλτράρονται από τα properties που έχουν επιλεγεί. Κάνοντας κλικ σε μία κλάση πηγαίνουμε σε νέο διάγραμμα που δείχνει τις υποκλάσεις της. Υπάρχει επιλογή back που πηγαίνει στο προηγούμενο επίπεδο και forward που πηγαίνει στο επόμενο (αν έχει προηγηθεί back). Κατά τη λειτουργία hover σε μια κλάση εμφανίζεται tooltip με τον αριθμό των instances της, τις υποκλάσεις της με τον αριθμό αντίστοιχο αριθμό instances, τα datatype properties με τον αριθμό εμφανίσεων, τη μέγιστη και την ελάχιστη τιμή τους, και τα object properties με τον αριθμό εμφανίσεών τους. Τα datatype και object properties αφορούν τριπλέτες των οποίων το subject ή το object ανήκει στη συγκεκριμένη κλάση. Όταν προστίθενται ή αφαιρούνται facets το treemap ξανασχεδιάζεται με τα νέα δεδομένα.
    - **Chart-Timeline:** Οπτικοποιούνται τα δεδομένα κάθε property που επελέγη από τα facets αφού φιλτραριστούν με βάση τις κλάσεις που έχουν επιλεγεί και ομαδοποιηθούν σε μια ιεραρχία με ομάδες σταθερού αριθμού στοιχείων. Κάθε property εμφανίζεται σαν διαφορετική σειρά – series (με διαφορετικό χρώμα) στο διάγραμμα. Αρχικά παρουσιάζονται οι ανώτερες ομάδες της ιεραρχίας που προκύπτει από το μοντέλο οπτικοποίησης. Κάνοντας κλικ σε μια ομάδα εμφανίζεται νέο διάγραμμα με τις αντίστοιχες υποομάδες του επόμενου επιπέδου. Αν η ομάδα που επελέγη επικαλύπτεται με ομάδες άλλων σειρών, στο επόμενο επίπεδο παρουσιάζονται και οι υποομάδες των άλλων σειρών που βρίσκονται μεταξύ του εύρους τιμών της ομάδας που κλίκαρε ο χρήστης. Σε περίπτωση που μια ομάδα βρίσκεται εν μέρει στο εύρος τιμών, η ομάδα χωρίζεται και εμφανίζεται μόνο το αποδεκτό τμήμα της. Στο κατώτερο επίπεδο της ιεραρχίας παρουσιάζονται τα δεδομένα των ομάδων που επελέγησαν σε bar chart για τα αριθμητικά δεδομένα (με δυνατότητα αλλαγής του τύπου του διαγράμματος σε area, line, spline, areaspline) και timeline για τις ημερομηνίες. Ο χρήστης μπορεί να μεγεθύνει τμήμα του διαγράμματος επιλέγοντας με το ποντίκι την περιοχή που επιθυμεί να μεγεθύνει. Για επιστροφή στο προηγούμενο επίπεδο υπάρχει επιλογή back καθώς και η επιλογή forward για μετάβαση σε μια επόμενη κατάσταση (αν έχει προηγηθεί back). Ακόμα, δίνεται δυνατότητα αφαίρεσης σειρών από το

διάγραμμα ή απόκρυψης και επανεμφάνισής τους για περιπτώσεις που το διάγραμμα περιέχει πολλά και αλληλοεπικαλυπτόμενα δεδομένα και επιθυμούμε να δούμε συγκεκριμένες σειρές με μεγαλύτερη λεπτομέρεια. Κατά τη λειτουργία hover σε μια ομάδα εμφανίζεται tooltip που περιλαμβάνει τον αριθμό στοιχείων της αντίστοιχης ομάδας, το ποσοστό τους σε σχέση με το συνολικό αριθμό στοιχείων, το εύρος της ομάδας (min-max) και ενδεικτικές τιμές δεδομένων (subjects) που περιέχει η ομάδα. Στα charts που απεικονίζουν αριθμητικά δεδομένα εμφανίζονται επιπλέον η μέση τιμή και η διακύμανση των δεδομένων που περιέχει η ομάδα. Ο χρήστης μπορεί επιπλέον να ρυθμίσει παραμέτρους που αφορούν την ομαδοποίηση των δεδομένων προς οπτικοποίηση όπως είναι ο αριθμός στοιχείων κάθε ομάδας σε ποσοστό σε σχέση με το συνολικό πλήθος στοιχείων (αφορά το κατώτερο επίπεδο), ο αριθμός επιπέδων ιεραρχίας, ο αριθμός ομάδων ανά επίπεδο σε ποσοστό του αριθμού ομάδων που περνούν από το ένα επίπεδο της ιεραρχίας στο επόμενο. Τέλος, στα charts δίνεται η δυνατότητα ομαδοποίησης των δεδομένων σε ομάδες σταθερού εύρους.

5

Σχεδίαση Συστήματος

Ακολουθεί η σχεδίαση του συστήματος.

5.1 Αρχιτεκτονική λογισμικού

Η αρχιτεκτονική του συστήματός μας βασίζεται στο σχεδιαστικό πρότυπο του Model View Controller και ακολουθεί μια client/server αρχιτεκτονική όπου ο client στέλνει requests μέσω ενός Web browser σε ένα JSP servlet. Στη συνέχεια ο servlet στέλνει request στο Java back-end το οποίο ανακτά και αποθηκεύει RDF δεδομένα από το TDB Store ή τη μνήμη, τα επεξεργάζεται και στέλνει τα απαραίτητα αντικείμενα πίσω στο servlet. Τέλος ο servlet κωδικοποιεί τα δεδομένα σε JSON μορφή και τα στέλνει στο front-end, το οποίο εκτελείται στον browser και παρουσιάζει τα αποτελέσματα στο χρήστη. Για τα requests χρησιμοποιήσαμε την τεχνολογία Ajax. Έτσι ο κώδικάς μας αποτελείται από τα εξής επιμέρους τμήματα:

- Front-end (View)

  Εκτελείται στο browser και είναι υπεύθυνο για την παρουσίαση του user interface και την αλληλεπίδραση του χρήστη με την εφαρμογή. Στέλνει ajax requests στο servlet και παρουσιάζει τις απαντήσεις μετά από την απαραίτητη επεξεργασία στον χρήστη.
- Servlet (Controller)

- Servlet: Λαμβάνει requests από το front-end και στέλνει τα απαραίτητα requests στο back-end. Επεξεργάζεται ή κωδικοποιεί τις απαντήσεις και στέλνει τα αποτελέσματα στο front-end.
- Series: Αντικείμενα της κλάσης αυτής αποτελούν τα δεδομένα των διαγραμμάτων chart και timeline του front-end.

- Back-end (Model)
  - LoadData: Φόρτωση RDF δεδομένων.
  - Statistics: Υπολογισμός στατιστικών.
  - Preprocessing: Προεπεξεργασία δεδομένων.
  - Grouping: Επιπλέον ομαδοποίηση και ιεραρχική οργάνωση των προς οπτικοποίηση δεδομένων και υπολογισμός των αντίστοιχων στατιστικών.
  - Visualization: Ανάκτηση και επεξεργασία των απαραίτητων δεδομένων για τις διάφορες μορφές οπτικοποίησης.
  - Utils: Χρήσιμες συναρτήσεις για την εκτέλεση των παραπάνω λειτουργιών.

5.1.1 Αναλυτική Περιγραφή

Παρακάτω περιγράφουμε συνοπτικά κάποιες από τις λειτουργίες και μεθόδους της εφαρμογής.

5.1.1.1 Front-End

- Index
  Παρέχει το interface σε HTML που βλέπει ο χρήστης.
- Javascript
  Χειρίζεται την εμφάνιση και την αλληλεπίδραση του χρήστη με την εφαρμογή μέσω ajax requests στο server, επεξεργασίας και παρουσίασης των αποτελεσμάτων τους
- css
  Εμπλουτίζει την εμφάνιση του user interface.

5.1.1.2 Servlet

Μέθοδοι:

- *public void init(ServletConfig config)*
  Αρχικοποιεί το context του Servlet και το path του αρχείου εισόδου σε περίπτωση που ο χρήστης ανεβάζει δικό του αρχείο.
- *protected void doGet(HttpServletRequest request, HttpServletResponse response)*
  Επεξεργάζεται requests από το front-end τύπου GET. Καλεί για το χειρισμό τους τη μέθοδο doPost().

- *protected void doPost(HttpServletRequest request, HttpServletResponse response)*
  Επεξεργάζεται requests από το front-end τύπου POST. Όπου χρειαστεί καλεί μεθόδους του Back-end και επιστρέφει τα αποτελέσματά τους ως response.

5.1.1.3 Back-end

5.1.1.3.1 LoadData

- *public boolean createModel()*
  Ανακτά δεδομένα από το TDB Store αν ήδη υπάρχουν, διαφορετικά δημιουργεί ένα TDB Store, διαβάζει τα δεδομένα εισόδου και τα αποθηκεύει.
  Διαβάζει την οντολογία που περιγράφει τα δεδομένα, εάν δίνεται.
  Εισάγει άλλες οντολογίες που αναφέρονται στο dataset.
  Πραγματοποιεί εξαγωγή συμπερασμάτων (inference) για τα δεδομένα.
  Επιστρέφει true αν τα δεδομένα ήδη υπήρχαν, false αν δημιούργησε το TDB Store.
- *public boolean createPreprocDataset()*
  Ανακτά τα δεδομένα της προεπεξεργασίας αν ήδη υπάρχουν διαφορετικά δημιουργεί ένα TDB Store στο οποίο θα αποθηκευτούν τα δεδομένα της προεπεξεργασίας που θα γίνει.
  Επιστρέφει true αν τα δεδομένα υπήρχαν, false αν δημιούργησε το TDB Store.
- *public void createMemModel()*
  Δημιουργεί ένα RDF μοντέλο στη μνήμη αντί για TDB Store.
  Διαβάζει τα δεδομένα εισόδου και την οντολογία που τα περιγράφει, αν δίνεται.
  Εισάγει άλλες οντολογίες που αναφέρονται στο dataset.
  Πραγματοποιεί εξαγωγή συμπερασμάτων (inference) για τα δεδομένα.
- *public void createPreprocMemDataset()*
  Δημιουργεί ένα RDF μοντέλο στη μνήμη στο οποίο θα αποθηκευτούν τα δεδομένα της προεπεξεργασίας.

5.1.1.3.2 Statistics

- *public void datasetStatistics(Model m, Dataset ds)*
  Υπολογίζει διάφορους στατιστικούς δείκτες για τα δεδομένα που περιέχει το RDF μοντέλο m και τους αποθηκεύει στο Dataset ds.
  Καλεί διάφορες βοηθητικές μεθόδους για τις παραπάνω λειτουργίες όπως την private void top5(Model m, HashMap map) που βρίσκει τα πέντε κλειδιά του map με το μεγαλύτερο value και τα αποθηκεύει στο μοντέλο m.

- Παρέχονται get μέθοδοι για ανάκτηση κάθε κατηγορίας στατιστικών δεικτών που υπολογίστηκαν.

5.1.1.3.3 Preprocessing

- *public HashMap preprocAll()*

  Μετρά τα instances κάθε κλάσης και υπολογίζει τον αρθμό instances των οποίων η κλάση δεν αναγνωρίστηκε. Τα αποθηκεύει σε ένα HashMap το οποίο επιστρέφει η μέθοδος.

  Για κάθε κλάση υπολογίζει τα object και datatype properties στα οποία εμφανίζεται μαζί με τον αντίστοιχο αριθμό εμφανίσεων.

  Τα παραπάνω αποθηκεύονται στο dataset που δημιουργήθηκε για το σκοπό αυτό σε Named Graphs με συγκεκριμένα ονόματα.

  Βρίσκει τα properties με object αριθμητική τιμή ή ημερομηνία και για το καθένα υπολογίζει τον αριθμό εμφανίσεών του, τις κλάσεις που έχει ως domain και τους τύπους δεδομένων (datatypes) που έχει ως range. Τα αποθηκεύει σε Named Graphs του dataset της προεπεξεργασίας.

  Ομαδοποιεί τις τριπλέτες του κάθε property με object αριθμό ή ημερομηνία με βάση την κλάση στην οποία ανήκει το subject τους και τις αποθηκεύει σε χωριστά Named Graphs του dataset της προεπεξεργασίας.

  Υπολογίζει το συνολικό αριθμό κλάσεων και τις 5 κλάσεις με τα περισσότερα Instances και τα αποθηκεύει μαζί με τα υπόλοιπα στατιστικά.
- *public void preprocClasses(HashMap instances)*

  Δέχεται σαν είσοδο ένα HashMap με τις κλάσεις του dataset και τον αντίστοιχο αριθμό instances της κάθε μίας.

  Υπολογίζει τις κλάσεις που βρίσκονται στην κορυφή της ιεραρχίας και τις αποθηκεύει στο αντίστοιχο Named Graph του dataset της προεπεξεργασίας μαζί με τον αριθμό instances της κάθε μίας.

  Για κάθε κλάση βρίσκει τις υποκλάσεις της και τις αποθηκεύει στο αντίστοιχο Named Graph του dataset της προεπεξεργασίας μαζί με τον αριθμό instances της κάθε μίας.

5.1.1.3.4 Grouping

- *public void createGroups(ArrayList classes, ArrayList numProperties, ArrayList dateProperties, double NumberOfElems, int hierarchy, double ratio, double fixedratio)*

Η μέθοδος αυτή ομαδοποιεί τις τριπλέτες με subject που ανήκει στις κλάσεις που δίνονται (classes) και predicate που ανήκει στα numProperties ή dateProperties που δίνονται ως παράμετροι με βάση την τιμή του object τους και τις οργανώνει σε μια ιεραρχία. Σαν παράμετροι δίνονται επίσης δείκτες που καθορίζουν τη μορφή της ομαδοποίησης.

Για κάθε property συνενώνει σε ένα RDF μοντέλο όλες τις τριπλέτες του μοντέλου της προεπεξεργασίας που το έχουν ως property.

Για τα αριθμητικά properties υπολογίζεται επιπλέον η μεγαλύτερη και η μικρότερη τιμή των τριπλετών που συνενώθηκαν.

Για κάθε property καλείται η μέθοδος CreateGroupsProp(int NumberOfElems, int hierarchy, double ratio, String property, double total).

Για τα αριθμητικά properties καλείται επιπλέον η μέθοδος CreateGroupsPropFixedRange(double min, double max, double range, int hierarchy, double ratio, String property, double total).

- *public void CreateGroupsProp(int NumberOfElems, int hierarchy, double ratio, String property, double total)*

  Χωρίζει τις τριπλέτες του property που δέχεται σαν παράμετρο σε ομάδες με σταθερό αριθμό στοιχείων, με βάση την τιμή του object τους. Για το σκοπό αυτό καλεί τη μέθοδο createGroupsForProperty(String property, int NumberOfElems, double total) η οποία φτιάχνει τις ομάδες στο κατώτερο επίπεδο της ιεραρχίας.

  Στη συνέχεια, για κάθε επόμενο επίπεδο της ιεραρχίας (ο συνολικός αριθμός επιπέδων δίνεται από την παράμετρο hierarchy) υπολογίζει τον αριθμό στοιχείων που θα έχουν οι ομάδες του από την παράμετρο ratio και τον αριθμό ομάδων του προηγούμενου επιπέδου και καλεί τη μέθοδο createGroupsForGroup(ArrayList<Group> GroupsOld, int numOfElems, int level, String property, double total, String fixed) η οποία ομαδοποιεί περαιτέρω τις ομάδες του προηγούμενου επιπέδου.

- *public void CreateGroupsPropFixedRange(double min, double max, double range, int hierarchy, double ratio, String property, double total)*

  Χωρίζει τις τριπλέτες του property που δέχεται σαν παράμετρο σε ομάδες σταθερού εύρους τιμών. Για το σκοπό αυτό καλεί τη μέθοδο createGroupsForPropertyFixedRange(String property, double lower, double upper, double range, double total) η οποία φτιάχνει τις ομάδες στο κατώτερο σημείο της ιεραρχίας.

  Στη συνέχεια, για κάθε επόμενο επίπεδο της ιεραρχίας υπολογίζει τον αριθμό στοιχείων που θα έχουν οι ομάδες του από την παράμετρο ratio και τον αριθμό ομάδων του προηγούμενου επιπέδου και καλεί τη μέθοδο

createGroupsForGroup(oldGroups,newNumOfElems,i,property, total, "FIXED") η οποία ομαδοποιεί περαιτέρω τις ομάδες του προηγούμενου επιπέδου.

- *protected ArrayList createGroupsForProperty(String property, int NumberOfElems, double total)*

  Δέχεται ως παραμέτρους την property της οποίας τις τριπλέτες θα ομαδοποιήσει, το συνολικό αριθμό των αντίστοιχω τριπλετών (total) και το ποσοστό του συνολικού αριθμού που θα συμπεριληφθεί σε κάθε ομάδα.

  Υπολογίζει το σταθερό αριθμό στοιχείων που θα περιέχει κάθε ομάδα από τις παραμέτρους NumberOfElems και total.

  Ομαδοποιεί τις τριπλέτες της συγκεκριμένης property σε ομάδες με τον αριθμό στοιχείων που υπολογίστηκε με βάση την τιμή του object τους, από το μικρότερο στο μεγαλύτερο. Για κάθε ομάδα υπολογίζει στατιστικούς δείκτες που περιγράφουν το περιεχόμενό της. Οι τριπλέτες κάθε ομάδας αποθηκεύονται σε ξεχωριστό Named Graph.

  Επιστρέφεται ArrayList με πληροφορίες για τις ομάδες που φτάχτηκαν.

- *protected ArrayList createGroupsForPropertyFixedRange(String property, double lower, double upper, double range, double total)*

  Δέχεται ως παραμέτρους την property τριπλέτες της οποίας θα ομαδοποιήσει, το συνολικό αριθμό των αντίστοιχων τριπλετών (total) καθώς και τη μέγιστη (upper) και ελάχιστη (lower) τιμή τους.

  Για κάθε ομάδα υπολογίζει το εύρος της και συμπεριλαμβάνει τις τριπλέτες των οποίων η τιμή του object βρίσκεται στο εύρος της συγκεκριμένης ομάδας. Υπολογίζει ακόμα στατιστικούς δείκτες που περιγράφουν το περιεχόμενό της. Οι τριπλέτες κάθε ομάδας αποθηκεύονται σε ξεχωριστό Named Graph.

  Επιστρέφεται ArrayList με πληροφορίες για τις ομάδες που φτιάχτηκαν.

- *protected ArrayList createGroupsForGroup(ArrayList<Group> GroupsOld, int numOfElems, int level, String property, double total, String fixed)*

  Δέχεται σαν παράμετρο μια λίστα με τις ομάδες του προηγούμενου επιπέδου, τις οποίες ομαδοποιεί περαιτέρω (GroupsOld) καθώς και τον αριθμό ομάδων του προηγούμενου επιπέδου που θα περιέχει κάθε νέα ομάδα.

  Εντάσσει κάθε ομάδα του προηγούμενου επιπέδου σε μια νέα ομάδα, υπολογίζει στατιστικά που περιγράφουν το περιεχόμενο των νέων ομάδων και αποθηκεύει τις αντίστοιχες πληροφορίες σε ένα Named Graph.

  Επιστρέφεται ArrayList με πληροφορίες για τις ομάδες που φτιάχτηκαν.

5.1.1.3.5 Visualization

Οι παρακάτω μέθοδοι παρέχουν τα δεδομένα για τη δημιουργία facets και διαγραμμάτων του front-end.

- *public ArrayList getSubClassesTree(String className, String sort)*
  Επιστρέφει τις υποκλάσεις της κλάσης className, που δίνεται ως παράμετρος μαζί με τον αριθμό instances τους, ταξινομημένες αλφαβητικά ή σε φθίνουσα σειρά ως προς τον αριθμό των instances τους, ανάλογα με την παράμετρο sort. Για το σκοπό αυτό κάνει query στο αντίστοιχο Named Graph. Τα δεδομένα που επιστρέφει χρησιμοποιούνται για τη δημιουργία του facet κλάσεων του front-end.
- *public ArrayList getProperties(String type, String sort)*
  Επιστρέφει τα properties τύπου αριθμού ή ημερομηνίας (type) από το αντίστοιχο Named Graph μαζί με τον αριθμό εμφανίσεών τους, το domain και το range τους, ταξινομημένα αλφαβητικά ή σε φθίνουσα σειρά ως προς τον αριθμό εμφανίσεών τους, ανάλογα με την παράμετρο sort. Τα δεδομένα που επιστρέφει χρησιμοποιούνται για τη δημιουργία των facets numeric και date properties του front-end.
- *public ArrayList getGroups(String property, int level, String fixed)*
  Επιστρέφει από τα αντίστοιχα Named Graphs τις ομάδες του ανώτερου επιπέδου της ιεραρχίας που αφορούν το property που δίνεται ως παράμετρος, μαζί με τα στατιστικά τους. Η παράμετρος fixed υποδηλώνει αν οι ζητούμενες ομάδες είναι σταθερού εύρους ή σταθερού αριθμού στοιχείων. Τα δεδομένα που επιστρέφονται χρησιμοποιούνται για το σχεδιασμό του ανώτερου επιπέδου του chart στο front-end.
- *public ArrayList getLowerLevel(String groupName, double min, double max, int level, String fixed)*
  Επιστρέφει από τα αντίστοιχα Named Graphs τις υποομάδες της ομάδας που δίνεται ως παράμετρος (groupName), των οποίων το εύρος βρίσκεται μεταξύ των τιμών min-max που δίνονται ως παράμετροι, μαζί με τα στατιστικά τους. Εάν μια υποομάδα βρίσκεται εν μέρει μεταξύ των τιμών min-max, η υποομάδα αυτή "κόβεται" και επιστρέφεται το αποδεκτό τμήμα. Η παράμετρος fixed υποδηλώνει αν οι ζητούμενες ομάδες είναι σταθερού εύρους ή σταθερού αριθμού στοιχείων. Τα δεδομένα που επιστρέφονται χρησιμοποιούνται για τη δημιουργία επόμενου επιπέδου του chart στο front-end.
- *public ArrayList getData(String groupName, double min, double max)*
  Επιστρέφει από τα αντίστοιχα Named Graphs τα τελικά δεδομένα του κατώτερου επιπέδου που περιλαμβάνει η ομάδα groupName που δίνεται ως παράμετρος και βρίσκονται μεταξύ των τιμών min και max. Τα δεδομένα που επιστρέφονται

χρησιμοποιούνται για το σχεδιασμό στο front-end του chart κατώτερου επιπέδου που περιλαμβάνει τα δεδομένα καθεαυτά.

- *public ArrayList getGroupsDates(String property, int level)*
  Ομοίως με getGroups αλλά για δεδομένα που αφορούν ημερομηνίες για το σχεδιασμό του timeline στο front-end.
- *public ArrayList getLowerLevelDate(String groupName, String min, String max, int level)*
  Ομοίως με getLowerLevelDate αλλά για δεδομένα που αφορούν ημερομηνίες για το σχεδιασμό του timeline στο front-end.
- *public ArrayList getDataDates(String groupName, String min, String max)*
  Ομοίως με getData αλλά για δεδομένα που αφορούν ημερομηνίες για το σχεδιασμό του timeline στο front-end.
- *public ArrayList treemap(ArrayList classes, ArrayList properties)*
  Φιλτράρει τις τριπλέτες των κλάσεων με βάση τα properties που δίνονται ως παράμετροι και υπολογίζει τα απαραίτητα στατιστικά.
  Επιστρέφει για κάθε κλάση που δόθηκε τον αριθμό instances της, τις υποκλάσεις της με τον αντίστοιχο αριθμό instances και τα datatype και object properties της με τα αντίστοιχα στατιστικά.
  Τα δεδομένα αυτά χρησιμοποιούνται από το front-end για το σχεδιασμό του ανώτερου επιπέδου της ιεραρχίας του treemap.
- *public ArrayList treemapNext(String clas, ArrayList properties)*
  Βρίσκει τις υποκλάσεις της κλάσης clas που δίνεται ως παράμετρος και φιλτράρει τις τριπλέτες τους με βάση τα properties που δίνονται, υπολογίζοντας τα απαραίτητα στατιστικά.
  Επιστρέφει τις υποκλάσεις μαζί με τον αριθμό instances τους, τις υποκλάσεις κάθε μίας με τον αντίστοιχο αριθμό instances, και τα datatype και object properties κάθε μίας με τα αντίστοιχα στατιστικά.
  Τα δεδομένα αυτά χρησιμοποιούνται από το front-end για το σχεδιασμό των επόμενων επιπέδων ιεραρχίας του treemap.

5.2 *Σχήμα Οντολογίας του Μοντέλου*

Κατά την προεπεξεργασία των δεδομένων και την εξαγωγή του μοντέλου οπτικοποίησης υπολογίζουμε επιπλέον δείκτες και στατιστικά. Για την περιγραφή τους ορίσαμε την παρακάτω οντολογία.

```
<!DOCTYPE rdf:RDF [
    <!ENTITY viz  "http://mynamespace#" >
     ]>
<rdf:RDF
    xmlns     ="&viz;"
    xmlns:vin ="&viz;"
    xml:base  ="&viz;"
    xmlns:owl ="http://www.w3.org/2002/07/owl#"
    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#">

    <owl:Class rdf:ID="Group"/>

    <owl:DatatypeProperty rdf:ID="Count">
        <rdfs:domain rdf:resource="http://www.w3.org/2002/07/owl#Thing" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="GroupCount">
        <rdfs:subPropertyOf rdf:resource="#Count" />
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="ClassCount">
        <rdfs:subPropertyOf rdf:resource="#Count" />
        <rdfs:domain rdf:resource="http://www.w3.org/2002/07/owl#Class" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="PropertyCount">
        <rdfs:subPropertyOf rdf:resource="#Count" />
        <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="dtPropertyCount">
        <rdfs:subPropertyOf rdf:resource="#PropertyCount" />
        <rdfs:domain
rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="objPropertyCount">
        <rdfs:subPropertyOf rdf:resource="#Count" />
        <rdfs:domain
rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>

    <owl:ObjectProperty rdf:ID="Domain">
        <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property" />
        <rdfs:range  rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="Range">
        <rdfs:domain rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-
ns#Property" />
        <rdfs:range  rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Datatype"/>
    </owl:ObjectProperty>
```

```
    <owl:ObjectProperty rdf:ID="belongsToGroup">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="#Group"/>
    </owl:ObjectProperty>

    <owl:DatatypeProperty rdf:ID="percent">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="mean">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="var">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="xsd:double"/>
    </owl:DatatypeProperty>

    <owl:ObjectProperty rdf:ID="subMin">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Resource"/>
    </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="subMax">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range  rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Resource"/>
    </owl:ObjectProperty>

    <owl:DatatypeProperty rdf:ID="min">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range>
            <owl:unionOf>
                    <rdf:li rdf:resource="xsd:string"/>
                    <rdf:li rdf:resource="xsd:date"/>
            </owl:unionOf>
        </rdfs:range>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="max">
        <rdfs:domain rdf:resource="#Group" />
        <rdfs:range>
            <owl:unionOf>
                    <rdf:li rdf:resource="xsd:string"/>
                    <rdf:li rdf:resource="xsd:date"/>
            </owl:unionOf>
        </rdfs:range>
    </owl:DatatypeProperty>

</rdf:RDF>
```

5.3 Υπολογισμός στατιστικών

Ο υπολογισμός στατιστικών γίνεται με βάση τον αλγόριθμο που περιγράφηκε στο (25) και παρουσιάζεται στην ενότητα 3.3, με ορισμένες τροποποιήσεις στα κριτήρια ώστε να ανταποκρίνονται στις απαιτήσεις της εφαρμογής μας. Διατρέχουμε όλες τις τριπλέτες του dataset μία φορά και ανάλογα με τα κριτήρια που πληρεί μια τριπλέτα, κρατάμε σε ορισμένους δείκτες και δομές που έχουμε ορίσει τις απαραίτητες πληροφορίες. Στο τέλος εφαρμόζουμε μια επιπλέον επεξεργασία όπου χρειάζεται και αποθηκεύουμε τα αποτελέσματα

στους αντίστοιχους named graphs ώστε να είναι κατηγοριοποιημένα και προσβάσιμα από την εφαρμογή.

Παρακάτω παρουσιάζουμε συνοπτικά τα στατιστικά κριτήρια και τον τρόπο υπολογισμού και αποθήκευσής τους.

Για κάθε τριπλέτα ?s ?p ?o του dataset:

- Συμπληρώνουμε τις παρακάτω δομές

Named Graphs		
Name	*Description*	*Filter - Action*
HashMap propM	key: properties που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	propM[?p, val++]
HashMap propDtM	key: datatype properties που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	if(isLiteral(?o)) then propDtM[?p, val++]
HashMap propObjM	key: object properties που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	if(isIRI(?o)) then propDtM[?p, val++]
HashMap vocabM	key: λεξιλόγια (vocabularies) που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	vocabM[namespace(?s), val++], vocabM[namespace(?p), val++], vocabM[namespace(?o), val++]
HashMap langM	key: γλώσσες (languages) που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	if(isLiteral(?o) then langM[language(?o), val++]
HashMap datatypesM	key: τύποι δεδομένων (datatypes) που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	if(isLiteral(?o) then datatypesM[type(?o), val++]
HashMap linksM	key: σύνδεσμοι μεταξύ λεξιλογίων, value: αντίστοιχος αριθμός εμφανίσεων	if(namespace(?s) != namespace(?o)) then linksM[namespace(?s)+namepace(?o), val++]
HashMap entitiesDist	key: entities (resources / IRIs) που χρησιμοποιούνται, value: αντίστοιχος αριθμός εμφανίσεων	if(isIRI(?s)) entitiesDistM[?s, val++] if(isIRI(?p)) entitiesDistM[?p, val++] if(isIRI(?o)) entitiesDistM[?o, val++]
HashMap outdegreeM	key: entities, value: αντίστοιχο outdegree (αριθμός outcoming properties - για κάθε entity, #triples όπου είναι subject)	outdegreeM[?s, val++]
HashMap indegreeM	key: entities, value: αντίστοιχο indegree (αριθμός incoming properties - για κάθε entity, #triples όπου είναι object)	indegreeM[?o, val++]
HashMap ppEM	key: entities, value: Set με τις properties των τριπλετών όπου εμφανίζεται η entity	ppEM[?s, Set.add(?p)] ppEM[?o, Set.add(?p)]

HashMap sameAs	key: λεξιλόγια που που εμφανίζονται σε sameAs triples, value: αντίστοιχος αριθμός εμφανίσεων	if(?p==owl:sameAs && namespace(?s) != namespace(?o)) then sameAs[namespace(?o), val++]

- Υπολογίζουμε τα top-5 στοιχεία (keys) με το μεγαλύτερο αριθμό εμφανίσεων (value) για τα HashMaps classM, vocabM, propM, propDtM, propObjM, indegreeM, outdegreeM, linksM, sameAsM, datatypesM, langM, entitiesDistM.
- Υπολογίζουμε τους παρακάτω δείκτες.

Named Graphs		
Name	*Description*	*Filter - Action*
triples = 0	αριθμός τριπλετών του dataset	triples++
triplesL = 0	αριθμός τριπλετών με literal object	if(isLiteral(?o)) triplesL++
triplesIRI = 0	αριθμός τριπλετών με IRI object	if(isIRI(?o)) triplesIRI++
schema = 0	αριθμός τριπλετών που περιγράφουν το σχήμα	if(namespace(?p))==rdf || rdfs || owl then schema++
sameAs = 0	αριθμός τριπλετών με predicate sameAs	if(?p==owl:sameAs) sameAs++
seeAlso = 0	αριθμός τριπλετών με predicate seeAlso	if(?p==owl:seeAlso) seeAlso++
blank = 0	αριθμός κενών κόμβων	if(isBlank(?s)) blank++ if(isBlank(?o)) blank++
blankS = 0	αριθμός κενών κόμβων ως subjects	if(isBlank(?s)) blank++
blankO = 0	αριθμός κενών κόμβων ως objects	if(isBlank(?o)) blank++
Classes = 0	αριθμός κλάσεων που χρησιμοποιούνται	size(ClassesM)
subClasses = 0	αριθμός υποκλάσεων που χρησιμοποιούνται	if(?p==rdfs:subclass || ?p==owl:subclass) subclasses++
avPPE = 0	μέσος όρος αριθμού properties που αντιστοιχούν σε κάθε entity	sum(ppEM.values)/size(ppEM)
maxPPE = 0	μέγιστος αριθμός properties που αντιστοιχούν σε κάθε entity	ppEM.key(max(ppEM.values))
minPPE = 0	ελάχιστος αριθμός properties που αντιστοιχούν σε κάθε entity	ppEM.key(min(ppEM.values))
vocab = 0	αριθμός λεξιλογίων (vocabularies) που χρησιμοποιούνται	vocab=size(vocabM)
prop = 0	αριθμός properties που χρησιμοποιούνται	prop=size(propM)

propDt = 0	αριθμός datatype properties που χρησιμοποιούνται	propDtM=size(propM)
propObj = 0	αριθμός object properties που χρησιμοποιούνται	propObj=size(propObjM)
links = 0	αριθμός συνδέσμων	links=size(linksM)
datatypes = 0	αριθμός τύπων δεδομένων (datatypes) που χρησιμοποιούνται	datatypes=size(datatypesM)
lang = 0	αριθμός γλωσσών που χρησιμοποιούνται	lang=size(datatypeM)

entitiesMentioned = 0	αριθμός entities (resources / IRIs) που εμφανίζονται	if(isResource(?s) entitiesMentioned++
		if(isResource(?p) entitiesMentioned++
		if(isResource(?o) entitiesMentioned++

entitiesDist = 0	αριθμός διακριτών entities	entitiesDist=size(entitiesDistM)
comment = 0	αριθμός σχολίων	if(?p==rdfs:comment) comment++
label = 0	αριθμός ετικετών (labels)	if(?p==rdfs:label) label++
avIndegree = 0	μέσος όρος αριθμού incoming properties (για κάθε instance, #triples όπου είναι object)	sum(indegreeM.values)/size(indegreeM)
avOutdegree = 0	μέσος όρος αριθμού outcoming properties (για κάθε instance, #triples όπου είναι subject)	sum(outdegreeM.values)/size(indegreeM)
equivalentClassV = 0	αριθμός ισοδύναμων κλάσεων που ανήκουν σε διαφορετικά λεξιλόγια	if(?p==owl:equivalentClass && namespace(?s)!=namespace(?o)) equivalentClassV++
equivalentPropertyV = 0	αριθμός ισοδύναμων properties που ανήκουν σε διαφορετικά λεξιλόγια	if(?p==owl:equivalentProperty && namespace(?s)!=namespace(?o)) equivalentPropertyV++
subclassOfV = 0	αριθμός ισοδύναμων υποκλάσεων που ανήκουν σε διαφορετικά λεξιλόγια	if(?p==rdfs:subclassOf && namespace(?s)!=namespace(?o)) subclassOfV++
subpropertyOfV = 0	αριθμός ισοδύναμων subproperties που ανήκουν σε διαφορετικά λεξιλόγια	if(?p==rdfs:subpropertyOf && namespace(?s)!=namespace(?o)) subpropertyOfV++
broadMatch = 0	αριθμός τριπλετών skos:broadMatch	if(?p==skos:broadMatch && namespace(?s)!=namespace(?o)) broadMatch++
narrowMatch = 0	αριθμός τριπλετών skos:narrowMatch	if(?p==skos:narrowMatch && namespace(?s)!=namespace(?o)) narrowMatch++
prov1 – prov10 = 0	συμβολίζουμε με prov1 – 10 10 λεξιλόγια για provenance metadata και υπολογίζουμε τον αριθμό εμφανίσεων του καθενός	if(namespace(?p)==$prov_i$) $prov_i$++

- Αποθηκεύουμε τα αποτελέσματα στους αντίστοιχους Named Graphs.

Named Graphs	
Name	*Description*
http://DatasetStatistics/metrics	Δείκτες με την αριθμητική τιμή τους που περιγράφουν συνολικά το dataset: triples, triplesL, triplesIRI, schema, sameAs, seeAlso, prop, propDt, propObj, blank, blankS, blankO, avIndegree, avOutdegree, classes, subclasses, entitiesMentioned, entitiesDist, comments, labels, vocab, lang, avPPE, minPPE, maxPPE, links
http://DatasetStatistics/t5class	5 κλάσεις με το μεγαλύτερο αριθμό instances
http://DatasetStatistics/t5voc	5 vocabularies με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5prop	5 properties με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5propDt	5 datatype prop με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5propObj	5 object prop με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5lang	5 languages με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5entities	5 entities με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5datatypes	5 datatypes με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5links	5 links με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/t5outdegree	5 entities με το μεγαλύτερο outdegree
http://DatasetStatistics/t5indegree	5 entities με το μεγαλύτερο indegree
http://DatasetStatistics/t5sameAs	5 sameAs vocabularies με το μεγαλύτερο αριθμό εμφανίσεων
http://DatasetStatistics/metaSC	Metadata που ανήκουν στο λεξιλόγιο Semantic Sitemap
http://DatasetStatistics/metaDC	Metadata που ανήκουν στο λεξιλόγιο Dublin Core
http://DatasetStatistics/metaVOID	Metadata που ανήκουν στο λεξιλόγιο Vocabulary of Interlinked Datasets (VOID)
http://DatasetStatistics/accessSC	Metadata που υποδηλώνουν access methods στο dataset και ανήκουν στο λεξιλόγιο Semantic Sitemap
http://DatasetStatistics/accessVOID	Metadata που υποδηλώνουν access methods στο dataset και ανήκουν στο λεξιλόγιο VOID
http://DatasetStatistics/licenceMeta	Metadata που παρέχουν πληροφορίες για το licence του dataset
http://DatasetStatistics/vocMappingsOWL	Vocabulary-level mappings του λεξιλογίου OWL
http://DatasetStatistics/vocMappingsRDFS	Vocabulary-level mappings του λεξιλογίου RDFS
http://DatasetStatistics/vocMappingsSKOS	Vocabulary-level mappings του λεξιλογίου SKOS
http://DatasetStatistics/instMappings	Instance-level mappings

Σε κάθε Named Graph που αφορά δείκτες και αριθμό εμφανίσεων αποθηκεύουμε τριπλέτες της μορφής: *resource NS+Count value (double)*. Για παράδειγμα:

http://DatasetStatistics/metrics		
sub	*pred*	*obj*
metricsName1	NS+Count	*value1 (double)*
metricsName2	NS+Count	*value2 (double)*
⋮	⋮	⋮

Σε κάθε Named Graph που αφορά metadata αποθηκεύουμε την τριπλέτα του dataset που ανήκει στην κατηγορία μεταδεδομένων που εξετάζουμε. Για παράδειγμα:

http://DatasetStatistics/accessVOID		
sub	*pred*	*obj*
dataset	void:datadump	*value1*
dataset	void:sparqlEndpoint	*value2*
dataset	void:uriLookupEndpoint	*value3*

5.4 Προεπεξεργασία

Κατά την προεπεξεργασία διατρέχουμε όλα τα δεδομένα – τριπλέτες μία φορά και για κάθε τριπλέτα εκτελούμε ορισμένες ενέργειες ανάλογα με τα κριτήρια που πληρεί, με την ίδια λογική του υπολογισμού στατιστικών. Στη συνέχεια εκτελούμε υπολογισμούς στα αποτελέσματα που προέκυψαν και οργανώνουμε – αποθηκεύουμε τα τελικά αποτελέσματα στους κατάλληλους Named Graphs. Αναλυτικότερα:

Ορίζουμε τις εξής δομές:

- HashMap countInst;
  Για κάθε κλάση αποθηκεύει τον αριθμό instances της.
  key: ClassName (String), value: instances (double)
- HashMap dataProp;
  Για κάθε κλάση διατηρεί τις datatype properties που της αντιστοιχούν (τριπλέτες με datatype property στις οποίες το subject ανήκει στη συγκεκριμένη κλάση).
  key: ClassName (String), value: HashMap (key: datatypePropertyName (String), value: αριθμός εμφανίσεων (double))

- HashMap objProp;

  Για κάθε κλάση διατηρεί τις object properties που της αντιστοιχούν (τριπλέτες με object property στις οποίες το subject ή το object ανήκει στη συγκεκριμένη κλάση).

  key: ClassName (String), value: HashMap (key: objectPropertyName (String), value: αριθμός εμφανίσεων (double))
- HashMap numeric, date;

  Λίστες με όλες τις properties αριθμητικού (numeric) ή τύπου ημερομηνίας (date) μαζί με τον αριθμό εμφανίσεων κάθε μίας.

  key: propertyName (String), value: αριθμός εμφανίσεων (double).
- HashMap numericDom, dateDom;

  Για κάθε property διατηρεί τις κλάσεις που αποτελούν το domain της.

  key: propertyName, value: Set με τις κλάσεις – domain
- HashMap numericRan, dateRan;

  Για κάθε property διατηρεί τους τύπους δεδομένων (datatypes) που αποτελούν το range της.

  key: propertyName, value: Set με τους τύπους δεδομένων – range

Για κάθε τριπλέτα s p o:

- Βρίσκουμε την κλάση στην οποία ανήκει το subject, έστω classS, και το object αν είναι resource, έστω classO.
- Αυξάνουμε τον αντίστοιχο μετρητή των instances για τις κλάσεις. Αν η κλάση δεν αναγνωρίστηκε, αυξάνουμε το μετρητή των "unrecognized" κλάσεων.
- Αν το object o είναι literal, προσθέτουμε το αντίστοιχο property p στη λίστα των Datatype Properties της κλάσης classS, και αυξάνουμε το μετρητή αριθμού εμφανίσεών του.
- Αν το object o είναι resource, προσθέτουμε το αντίστοιχο property p στη λίστα των Datatype Properties της κλάσης classS και της κλάσης classO, και αυξάνουμε το μετρητή αριθμού εμφανίσεών του.
- Βρίσκουμε το datatype του object o.
- Εάν το datatype του object o είναι αριθμητικού τύπου:
  - Κατατάσσουμε το property στα numeric properties.
  - Αυξάνουμε το μετρητή του αντίστοιχου property p.
  - Προσθέτουμε το datatype στο range του αντίστοιχου property p.
  - Προσθέτουμε την κλάση στην οποία ανήκει το subject στο domain του property p ("unrecognized" αν δεν αναγνωρίστηκε).
  - Προσθέτουμε την τρέχουσα τριπλέτα s (sub) p (propName) o (obj) στο Named Graph που αντιστοιχεί στην κλάση του subject s και στο property p.

- Εάν το datatype του object o είναι τύπου ημερομηνίας:
    - Κατατάσσουμε το property στα date properties.
    - Αυξάνουμε το μετρητή του αντίστοιχου property p.
    - Προσθέτουμε το datatype στο range του αντίστοιχου property p.
    - Προσθέτουμε την κλάση στην οποία ανήκει το subject στο domain του property p ("unrecognized" αν δεν αναγνωρίστηκε).
    - Προσθέτουμε την τρέχουσα τριπλέτα s (sub) p (propName) o (obj) στο Named Graph που αντιστοιχεί στην κλάση του subject s και στο property p.

propName*NS*className

sub	*pred*	*obj*
sub1	propName	obj1
sub2	propName	obj2
⋮	⋮	⋮

Αποθηκεύουμε τα στοιχεία που υπολογίστηκαν στα αντίστοιχα Named Graphs:

- Για κάθε κλάση δημιουργούμε ένα ξεχωριστό Named Graph με το πρόθεμα *http://DatasetStatistics/classes/stats/* + Όνομα της κλάσης (className). Χρησιμοποιώντας προκαθορισμένα properties που έχουμε ορίσει στην οντολογία μας, αποθηκεύουμε
    - Το όνομα της κλάσης *(className)* και τον αριθμό instances της.
    - Όλες τις datatype properties (*dtPropName)* που αντιστοιχούν στην κλάση μαζί με τον αριθμό εμφανίσεων της κάθε μίας.
    - Όλες τις object properties (*objPropName)* που αντιστοιχούν στην κλάση μαζί με τον αριθμό εμφανίσεων της κάθε μίας.

http://DatasetStatistics/classes/stats/className

sub	*pred*	*obj*
className	NS+ClassCount	*value1 (double)*
dtPropName1	NS+dtPropCount	*value2 (double)*
⋮	⋮	⋮
objPropName1	NS+objPropCount	*value3 (double)*
⋮	⋮	⋮

- Βρίσκουμε τις κλάσεις που βρίσκονται στην κορυφή της ιεραρχίας κλάσεων *(className)* και τις αποθηκεύουμε στο Named Graph *http://DatasetStatistics/classes/rootClasses* μαζί με τον αντίστοιχο αριθμό instances.

http://DatasetStatistics/classes/rootClasses		
sub	*pred*	*obj*
className1	NS+ClassCount	*value1 (double)*
className2	NS+ClassCount	*value2 (double)*
⋮	⋮	⋮

- Για κάθε κλάση δημιουργούμε ένα ξεχωριστό Named Graph με το πρόθεμα *http://DatasetStatistics/classes/* + Όνομα της κλάσης (className). Βρίσκουμε τις υποκλάσεις της *(subclassName)* και τις αποθηκεύουμε μαζί με τον αριθμό instances της κάθε μίας.

http://DatasetStatistics/classes/className		
sub	*pred*	*obj*
subclassName1	NS+ClassCount	*value1 (double)*
subclassName2	NS+ClassCount	*value2 (double)*
⋮	⋮	⋮

- Δημιουργούμε ένα Named Graph με όνομα *http://DatasetStatistics/properties/numeric* στο οποίο αποθηκεύουμε όλες τις numeric properties και ένα με όνομα *http://DatasetStatistics/properties/date* για τις date properties. Για κάθε μία από αυτές *(propName)* αποθηκεύουμε τον αριθμό εμφανίσεών της, όλες τις κλάσεις που αποτελούν το domain της *(Class)*, και όλους τους τύπους δεδομένων που αποτελούν το range της *(Datatype)*.

http://DatasetStatistics/properties/numeric		
sub	*pred*	*obj*
propName1	NS+PropertyCount	*value1 (double)*
propName1	NS+Domain	*Class1*
⋮	⋮	⋮
propName1	NS+Range	*Datatype1*
⋮	⋮	⋮
propName2	NS+PropertyCount	*value2 (double)*
⋮	⋮	⋮

http://DatasetStatistics/properties/date		
sub	*pred*	*obj*
propName1	NS+PropertyCount	*value1 (double)*
propName1	NS+Domain	*Class1*
⋮	⋮	⋮
propName1	NS+Range	*Datatype1*
⋮	⋮	⋮
propName2	NS+PropertyCount	*value2 (double)*
⋮	⋮	⋮

Στον παρακάτω πίνακα φαίνονται όλοι οι Named Graphs που προκύπτουν κατά την προεπεξεργασία των δεδομένων.

Named Graphs	
Name	*Description*
propName+NS+*className*	Περιέχει τρίπλες (s, p, o) του dataset με property p=*propName* και s:typeOf(*className*)
⋮	
http://DatasetStatistics/classes/stats/*className*	Περιλαμβάνει στατιστικά που αφορούν την κλάση *className* πχ. αριθμό instances, datatype και object properties κλπ.
⋮	
http://DatasetStatistics/classes/rootClasses	Περιλαμβάνει τις κλάσεις στην κορυφή της ιεραρχίας κλάσεων με τον αριθμό instances τους.
http://DatasetStatistics/classes/*className*	Περιλαμβάνει τις υποκλάσεις της κλάσης *className* με τον αριθμό instances τους.
⋮	
http://DatasetStatistics/properties/numeric	Περιλαμβάνει τις numeric properties με πληροφορίες για τον αριθμό εμφανίσεων, domain, range κάθε μίας.
http://DatasetStatistics/properties/date	Περιλαμβάνει τις date properties με πληροφορίες για τον αριθμό εμφανίσεων, domain, range κάθε μίας.

5.5 *Ομαδοποίηση δεδομένων και Ιεραρχική οργάνωση – Μοντέλο οπτικοποίησης*

Όπως περιγράφηκε στην ενότητα 5.2.3.4, προκειμένου να γίνει η οπτικοποίηση, συγκεντρώνονται τα δεδομένα προς οπτικοποίηση από τα Named Graphs που προέκυψαν κατά την προεπεξεργασία, ομαδοποιούνται και οργανώνονται σε μια ιεραρχία. Ως παράμετροι για την ομαδοποίηση δίνονται ο αριθμός επιπέδων της ιεραρχίας (hierarchy), το ποσοστό των συνολικών δεδομένων που θα συμπεριληφθεί σε κάθε ομάδα του κατώτερου επιπέδου (numberOfElements), και ο αριθμός των ομάδων κάθε επιπέδου ως ποσοστό του αριθμού ομάδων του προηγούμενου επιπέδου. Η ομαδοποίηση σε κάθε επίπεδο γίνεται ως εξής:

- LEVEL 0-1

  Variable Range

  Στην περίπτωση αυτή οι τριπλέτες των προς οπτικοποίηση δεδομένων ομαδοποιούνται με βάση την τιμή του object τους σε ομάδες με σταθερό αριθμό στοιχείων (και κατ'επέκταση με μεταβλητό εύρος). Η ομαδοποίηση εφαρμόζεται για κάθε property ξεχωριστά, ακολουθώντας τα παρακάτω βήματα:

  - Υπολογίζουμε τον αριθμό στοιχείων n κάθε ομάδας:
    n=συνολικός αριθμός στοιχείων * numberOfElems (ποσοστό)
  - Οι τριπλέτες των δεδομένων ταξινομούνται σε αύξουσα σειρά ως προς την τιμή του object τους.
  - Οι n πρώτες τριπλέτες κατατάσσονται στην $1^{η}$ ομάδα, οι επόμενες n στη $2^{η}$ κ.ο.κ.
  - Κάθε ομάδα αναπαρίσταται με ένα ξεχωριστό Named Graph που περιέχει τις τριπλέτες που ανήκουν στην ομάδα. Το όνομα του named graph αποτελείται από το όνομα του property που αφορά *(propName)* + "Group/" + έναν αριθμό που δείχνει σε ποια ομάδα του property αναφέρεται *(index)* πχ http://dbpedia.org/ontology/areaTotal/Group2.

propName*/Group*index		
sub	*pred*	*obj*
sub1	pred1	obj1
sub2	pred2	obj2
⋮	⋮	⋮

Fixed Range

Στην περίπτωση αυτή οι τριπλέτες των προς οπτικοποίηση δεδομένων ομαδοποιούνται με βάση την τιμή του object τους σε ομάδες με σταθερό εύρος τιμών (και κατ' επέκταση με μεταβλητό αριθμό στοιχείων). Η ομαδοποίηση εφαρμόζεται για κάθε property ξεχωριστά, ακολουθώντας τα παρακάτω βήματα:

- Υπολογίζουμε το εύρος τιμών range κάθε ομάδας:
  range=(maxObject – minObject) * fixedRatio (ποσοστό)
  όπου maxObject και minObject η μέγιστη και η ελάχιστη αντίστοιχα τιμή των objects των τριπλετών του property που ομαδοποιείται.
- Οι τριπλέτες των δεδομένων ταξινομούνται σε αύξουσα σειρά ως προς την τιμή του object τους.
- Υπολογίζουμε τα όρια της $1^{ης}$ ομάδας [min, max] όπου min = minObject, max = minObject + range.
- Όσες τριπλέτες έχουν object < min εντάσσονται στην πρώτη ομάδα.
- Για κάθε επόμενη ομάδα, υπολογίζουμε τα όριά της (min = maxPrevious, max = maxPrevious + range) και εντάσσουμε τις τριπλέτες με object < max.
- Κάθε ομάδα αναπαρίσταται με ένα ξεχωριστό Named Graph που περιέχει τις τριπλέτες που ανήκουν στην ομάδα. Το όνομα του named graph αποτελείται από το όνομα του property που αφορά *(propName)* + "Group/" + έναν αριθμό που δείχνει σε ποια ομάδα του property αναφέρεται *(index)* + "FIXED" πχ. http://dbpedia.org/ontology/areaTotal/Group2FIXED.

***propName*/Group*index*FIXED**		
sub	*pred*	*obj*
sub1	pred1	obj1
sub2	pred2	obj2
⋮	⋮	⋮

- Και στις δύο περιπτώσεις (fixed, variable), για κάθε ομάδα που σχηματίστηκε υπολογίζουμε
  - Τον αριθμό στοιχείων-τριπλετών που περιέχει (GroupCount).
  - Το ποσοστό στοιχείων που περιέχει η ομάδα σε σχέση με το συνολικό αριθμό στοιχείων όλων των ομάδων του συγκεκριμένου property (percent).
  - Το εύρος της ομάδας, δηλαδή τη μικρότερη και τη μεγαλύτερη τιμή του object των τριπλετών που περιέχει η ομάδα (min, max).
  - Το subject της τριπλέτας με τη μικρότερη τιμή του object (subMin).

- Το subject της τριπλέτας με τη μεγαλύτερη τιμή του object (subMax).
- Αν έχουμε αριθμητικά δεδομένα, υπολογίζουμε το μέσο όρο και τη διακύμανση των objects των τριπλετών που περιέχει η ομάδα (mean, var).

- Σε όλα τα επίπεδα διατηρούμε ένα Named Graph που λειτουργεί ως index του επιπέδου και αποθηκεύει πληροφορίες για όλες τις ομάδες του επιπέδου (για όλα τα properties), με όνομα http://GroupsIndex/LEVEL + τρέχον επίπεδο. Για τις ομάδες με fixed range διατηρούμε ένα όμοιο named graph με όνομα http://GroupsIndex/LEVEL + τρέχον επίπεδο + "FIXED" πχ. http://GroupsIndex/LEVEL1FIXED.

http://GroupsIndex/LEVELlevel		
sub	*pred*	*obj*
groupName1	NS+GroupCount	*value1 (double)*
groupName1	NS+min	*value2 (double/date)*
groupName1	NS+max	*value3 (double/date)*
groupName1	NS+subMin	*resource1*
groupName1	NS+subMax	*resource2*
groupName1	NS+percent	*value4 (double)*
groupName1	NS+mean	*value5 (double)*
groupName1	NS+var	*value6 (double)*
groupName1	NS+belongsToGroup	*groupFromPrevLevel*
groupName2	NS+GroupCount	*value7 (double)*
⋮	⋮	⋮

- LEVEL 1 - … N

  Οι ομάδες που σχηματίστηκαν από τα αρχικά δεδομένα ομαδοποιούνται περαιτέρω για να σχηματίσουμε μια ιεραρχία. Κάθε ομάδα του νέου επιπέδου περιέχει έναν αριθμό ομάδων από το προηγούμενο επίπεδο. Δεν περιέχει τις τριπλέτες καθεαυτές που ανήκουν στη νέα ομάδα, αλλά τις ομάδες του προηγούμενου επιπέδου που ανήκουν στη νέα. Τα επίπεδα της ιεραρχίας (N) δίνονται από την παράμετρο hierarchy. Η ομαδοποίηση γίνεται ως εξής:
  - Υπολογίζεται ο αριθμός ομάδων που θα υπάρχουν στο επόμενο επίπεδο: newNumberOfGroups = oldNumberOfGroups * ratio.
  - Υπολογίζεται ο αριθμός στοιχείων-ομάδων που θα περιέχει κάθε νέα ομάδα: newNumberOfElements = oldNumberOfGroups / newNumberOfGroups

Πχ. Έστω ότι σε κάθε επίπεδο θέλουμε να έχουμε το μισό αριθμό ομάδων σε σχέση με το προηγούμενο, δηλαδή ratio = 0,5. Έστω ότι στο επίπεδο n-1 έχουμε 10 ομάδες και θέλουμε να υπολογίσουμε τις ομάδες του ανώτερου επιπέδου n. Στο επίπεδο n θα έχουμε 6*0,5=3 ομάδες. Άρα κάθε ομάδα του επιπέδου n θα έχει 6/3=2 στοιχεία-ομάδες του προηγούμενου επιπέδου.

- Διατρέχουμε τις ομάδες του προηγούμενου επιπέδου με τη σειρά από το μικρότερο εύρος προς το μεγαλύτερο.
- Οι newNumberOfElements πρώτες ομάδες μπαίνουν στην $1^{η}$ ομάδα, οι επόμενες newNumberOfElements στη $2^{η}$ κ.ο.κ.
- Το εύρος κάθε νέας ομάδας [minNew, maxNew] προκύπτει από το εύρος των ομάδων που περιέχει.

  Συνεχίζοντας το παράδειγμά μας, έστω ότι στο επίπεδο n-1 έχουμε τις εξής ομάδες:

  propGroup1LEVELn-1 [0, 10] propGroup2LEVELn-1 [10, 20]
  propGroup3LEVELn-1 [20, 30] propGroup4LEVELn-1 [30, 40]
  propGroup5LEVELn-1 [40, 50] propGroup6LEVELn-1 [50, 60]

  Στο επίπεδο n κάθε ομάδα θα περιλαμβάνει δύο ομάδες του προηγούμενου επιπέδου. Έτσι έχουμε:

  propGroup1LEVELn [0, 20] propGroup2LEVELn [20, 40]
  propGroup3LEVELn [40, 60]
- Ο αριθμός στοιχείων κάθε νέας ομάδας υπολογίζεται αθροίζοντας τους αριθμούς στοιχείων των ομάδων που περιέχει.
- Το ποσοστό στοιχείων της ομάδας σε σχέση με τα συνολικά στοιχεία υπολογίζεται διαιρώντας το νέο αριθμό στοιχείων που υπολογίστηκε προηγουμένως με το συνολικό αριθμό στοιχείων.
- Τα minSubject, maxObject κάθε νέας ομάδας υπολογίζονται ως το minSubject της μικρότερης ομάδας και το maxSubject της μεγαλύτερης ομάδας αντίστοιχα που περιλαμβάνει.
- Το όνομα κάθε ομάδας προκύπτει ως: όνομα του property που αφορά *(propName)* + "Group/" + έναν αριθμό που δείχνει σε ποια ομάδα του property αναφέρεται *(index)* + "LEVEL" + τρέχον επίπεδο *(level)* πχ. http://dbpedia.org/ontology/areaTotal/Group2LEVEL3.
- Όσον αφορά στα αριθμητικά δεδομένα, ο μέσος όρος και η διακύμανση των τιμών για κάθε νέα ομάδα υπολογίζονται γνωρίζοντας τους δείκτες αυτούς και το πλήθος των στοιχείων στις ομάδες που περιλαμβάνουν.
- Αποθηκεύουμε τις πληροφορίες αυτές στο named graph που αντιστοιχεί στο επίπεδο που βρισκόμαστε.

- Αποθηκεύουμε επίσης στο Named Graph τις ομάδες του προηγούμενου επιπέδου που περιλαμβάνει κάθε νέα ομάδα.

Στον παρακάτω πίνακα φαίνονται όλοι οι Named Graphs που προκύπτουν κατά την ομαδοποίηση των δεδομένων:

Named Graphs	
Name	*Description*
propName/Group*index*	Περιέχει τρίπλες του dataset που ανήκουν στο property *propName* και στην ομάδα με αριθμό *index*– κατώτερο επίπεδο
⋮	
propName/Group*index*FIXED	Περιέχει τρίπλες του dataset που ανήκουν στο property *propName* και στην ομάδα με αριθμό *index*– κατώτερο επίπεδο – ομάδες σταθερού εύρους
⋮	
http://GroupsIndex/LEVEL*level*	Περιέχει πληροφορίες για όλες τις ομάδες του επιπέδου *level*
⋮	
http://GroupsIndex/LEVEL*level*FIXED	Περιέχει πληροφορίες για όλες τις ομάδες σταθερού εύρους του επιπέδου *level*
⋮	

6

Υλοποίηση

6.1 Πλατφόρμες και προγραμματιστικά εργαλεία

6.1.1 Πλατφόρμες ανάπτυξης

Η ανάπτυξη της εφαρμογής έγινε σε περιβάλλον Ubuntu 13.04 με χρήση του Java JDK 1.7. Η εφαρμογή αναπτύχθηκε στο NetBeans IDE 7.3.1 με Java EE 6 Web και server Apache Tomcat 7.0.34.0.

6.1.2 Εργαλεία – Βιβλιοθήκες

6.1.2.1 Apache Jena Framework

Java framework που περιλαμβάνει εργαλεία και βιβλιοθήκες σε Java για την ανάπτυξη εφαρμογών σημασιολογικού ιστού και διασυνδεδεμένων δεδομένων. Παρέχει μεθόδους για αποδοτική αποθήκευση μεγάλου αριθμού RDF δεδομένων στο δίσκο, την αναπαράσταση και διαχείριση RDF Γράφων και την εκτέλεση SPARQL ερωτημάτων σε αυτούς.

6.1.2.2 Google Gson

Java βιβλιοθήκη που σειριοποιεί και αποσειριοποιεί δεδομένα σε / από JSON μορφή.

6.1.2.3 JavaScript Libraries

- JQuery 1.9.1
- Kendo UI για την υλοποίηση του User Interface
- JQuery Form για την αποστολή φόρμας στο server
- HighCharts για το σχεδιασμό Charts και Timelines
- Google Charts για το σχεδιασμό Treemap

6.1.3 Απαιτήσεις Εφαρμογής

Η εφαρμογή εκτελείται σε όλους τους σύγχρονους browsers με ενεργοποιημένη τη JavaScript και τα Cookies.

6.2 Κωδικοποίηση αρχείων

Το σύστημα χρησιμοποιεί RDF αρχεία εισόδου με τις παρακάτω κωδικοποιήσεις. Τα αρχεία αυτά διαβάζονται με εργαλεία του Apache Jena Framework και μετατρέπονται σε RDF Γράφους.

6.2.1 RDF/XML (.rdf)

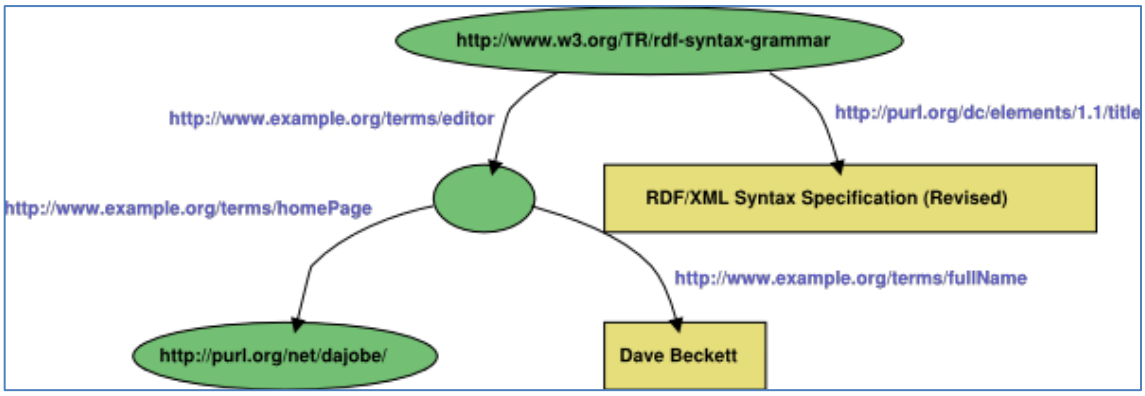
Τα αρχεία αυτού του τύπου περιέχουν RDF Γράφους κωδικοποιημένους με XML σύνταξη. Για να κωδικοποιηθεί ένας RDF γράφος σε XML πρέπει οι κόμβοι (subjects, objects) και οι ακμές (predicates) να αναπαρασταθούν με XML όρους. Το subject μπορεί να είναι ένα URI ή κενός κόμβος, το predicate ένα URI, και το object ένα URI, κενός κόμβος ή literal. Η σύνταξη RDF/XML χρησιμοποιεί τα QNames της XML για αναπαράσταση των RDF URIs. Το URI των subjects και objects μπορεί επίσης να αποθηκευτεί σαν τιμή από XML attributes. Τα RDF Literals γίνονται είτε κείμενο σε ένα XML element είτε τιμές XML attributes.

Ένας γράφος αποτελείται από πολλά μονοπάτια της μορφής:

κόμβος -> ακμή-predicate -> κόμβος -> ακμή-predicate …

Στη σύνταξη RDF/XML τα μονοπάτια αυτά μετατρέπονται σε ακολουθίες εμφωλευμένων XML elements που εναλλάσσονται μεταξύ elements για κόμβους και predicate ακμές. Συγκεκριμένα ο κόμβος που βρίσκεται στην αρχή της ακολουθίας αποτελεί το εξωτερικό element, η επόμενη predicate ακμή γίνεται παιδί του προηγούμενου element κ.ο.κ.

Παράδειγμα 1 Κωδικοποίηση RDF Graph σε RDF/XML (Πηγή: http://www.w3.org/TR/rdf-syntax-grammar/#figure1)




```
<rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
  <ex:editor>
    <rdf:Description>
      <ex:homePage>
        <rdf:Description rdf:about="http://purl.org/net/dajobe/">
        </rdf:Description>
      </ex:homePage>
    </rdf:Description>
  </ex:editor>
</rdf:Description>

<rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
  <ex:editor>
    <rdf:Description>
      <ex:fullName>Dave Beckett</ex:fullName>
    </rdf:Description>
  </ex:editor>
</rdf:Description>

<rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar">
  <dc:title>RDF/XML Syntax Specification (Revised)</dc:title>
</rdf:Description>
```

6.2.2 Turtle (.ttl)

Τα αρχεία Turtle κωδικοποιούν RDF Γράφους σε μια ειδική μορφή κειμένου. Ένα απλό RDF Statement (τριπλέτα) αναπαρίσταται με το subject, predicate και object χωρισμένα με whitespaces και τερματίζεται με μια τελεία. Το subject μπορεί να είναι ένα URI ή κενός κόμβος, το predicate ένα URI, και το object ένα URI, κενός κόμβος ή literal. Τα URIs μπορούν να γραφούν ως έχουν (absolute ή relative) ή με χρήση προθέματος (Prefix).

- Τα absolute και relative URIs περικλείονται από τους χαρακτήρες “<” και “>”.
- Τα literals αναπαρίστανται με την τιμή τους σε εισαγωγικά, και προαιρετικά ακολουθούμενα από τον τύπο τους.
- Οι κενοί κόμβοι αναπαρίστανται με “_:” ακολουθούμενο από ένα label.

Η σύνταξη Turtle παρέχει αρκετές συντομεύσεις για την αναπαράσταση πολλών τριπλετών μαζί. Οι συντομεύσεις αφορούν για παράδειγμα τριπλέτες με ίδιο subject και object και διαφορετικά predicates (predicate lists), object lists, συλλογές από RDF nodes, εμφώλευση κενών κόμβων κλπ.

Παράδειγμα 2 Turtle syntax (Πηγή: http://www.w3.org/TR/turtle/)

```
@base <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix rel: <http://www.perceive.net/schemas/relationship/> .

<#green-goblin>
    rel:enemyOf <#spiderman> ;
    a foaf:Person ;    # in the context of the Marvel universe
    foaf:name "Green Goblin" .

<#spiderman>
    rel:enemyOf <#green-goblin> ;
    a foaf:Person ;
    foaf:name "Spiderman", "Человек-паук"@ru .
```

6.2.3 N-Triples (.nt)

Η σύνταξη αυτή αποτελεί μια απλοποίηση της Turtle σύνταξης, καθώς επιτρέπει τη χρήση μόνο απλών RDF προτάσεων χωρίς συντομεύσεις. Σε κάθε γραμμή του αρχείου υπάρχει μία πρόταση με το subject, predicate και object χωρισμένα με whitespaces και τερματίζεται με μια τελεία. Οι κανόνες σύνταξης των URIs, literals και blank nodes ακολουθούν αυτούς της Turtle. Λόγω της απουσίας συντομεύσεων που κάνουν τη σύνταξη πιο πολύπλοκη, τα αρχεία n-triples είναι πιο εύκολο να διαβαστούν και να γραφούν από λογισμικό.

Παράδειγμα 3 Κωδικοποίηση RDF Graph σε RDF/XML (Πηγή: http://www.w3.org/2001/sw/RDFCore/ntriples/)

```
<http://www.w3.org/2001/sw/RDFCore/ntriples/>
<http://purl.org/dc/elements/1.1/creator> "Dave Beckett" .
<http://www.w3.org/2001/sw/RDFCore/ntriples/>
<http://purl.org/dc/elements/1.1/creator> "Art Barstow" .
<http://www.w3.org/2001/sw/RDFCore/ntriples/>
<http://purl.org/dc/elements/1.1/publisher> <http://www.w3.org/> .
```

6.2.4 N3 (.n3)

Η σύνταξη Turtle και N-triples αποτελούν απλοποιημένες μορφές της N3 σύνταξης. Οι Turtle και N-triples απλά σειριοποιούν RDF Γράφους ενώ η N3 παρέχει επιπλέον δυνατότητες όπως υποστήριξη RDF κανόνων (RDF-based rules).

7

Παρουσίαση Εφαρμογής

Ακολουθεί η παρουσίαση της εφαρμογής και η αξιολόγηση του συστήματος.

7.1 Μεθοδολογία

Η παρουσίαση και αξιολόγηση πραγματοποιείται με τη χρήση του παρακάτω σεναρίου λειτουργίας.

- Εισαγωγή δεδομένων στην εφαρμογή από ήδη φορτωμένα δεδομένα, από αρχείο, από URL.
- Απόκρυψη, εμφάνιση, ταξινόμηση του περιεχομένου των facets.
- Εμφάνιση στατιστικών.
- Εμφάνιση πληροφοριών του dataset.
- Σχεδιασμός και πλοήγηση στα διαγράμματα (chart, timeline, treemap).

Ως test data χρησιμοποιήσαμε ένα υποσύνολο δεδομένων της DBpedia, η οποία αποτελεί ένα από τα πιο διαδεδομένα linked datasets με πλήθος διαφορετικού τύπου δεδομένα, και χρησιμοποιείται ευρέως από παρόμοια εργαλεία. Χρησιμοποιήσαμε ακόμα Linked Statistics datasets καθώς περιλαμβάνουν πλήθος αριθμητικών δεδομένων και ημερομηνιών που μπορούν να οπτικοποιηθούν από την εφαρμογή μας.

Η αξιολόγηση της ευχρηστίας του User Interface πραγματοποιήθηκε με συλλογή feedback από χρήστες διαφορετικού επιπέδου γνώσης των τεχνολογιών διασυνδεδεμένων δεδομένων, και επανασχεδιασμό του User Interface λαμβάνοντας υπόψη τις παρατηρήσεις τους.

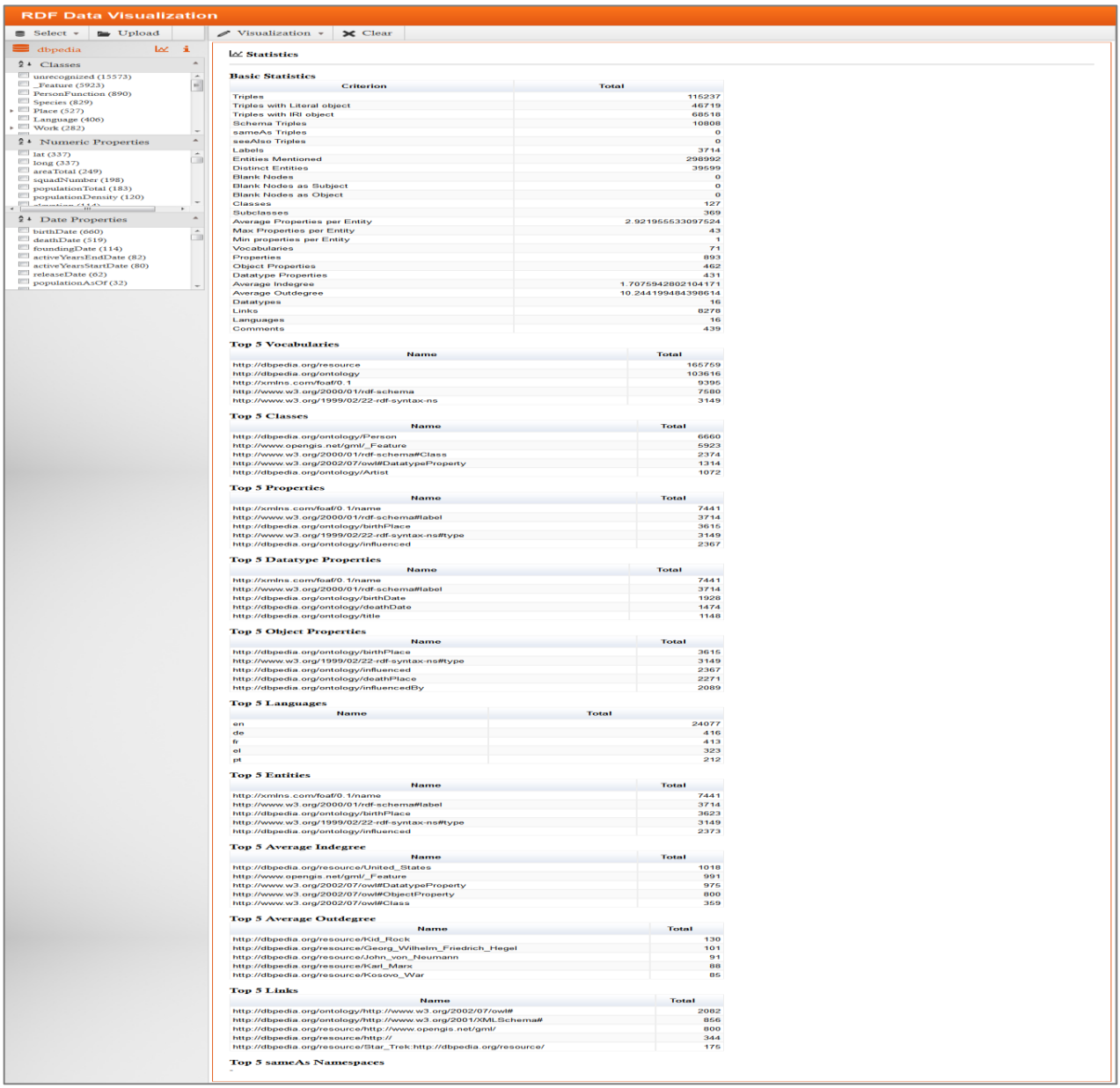
7.2 *Αναλυτική παρουσίαση – Εγχειρίδιο χρήσης*

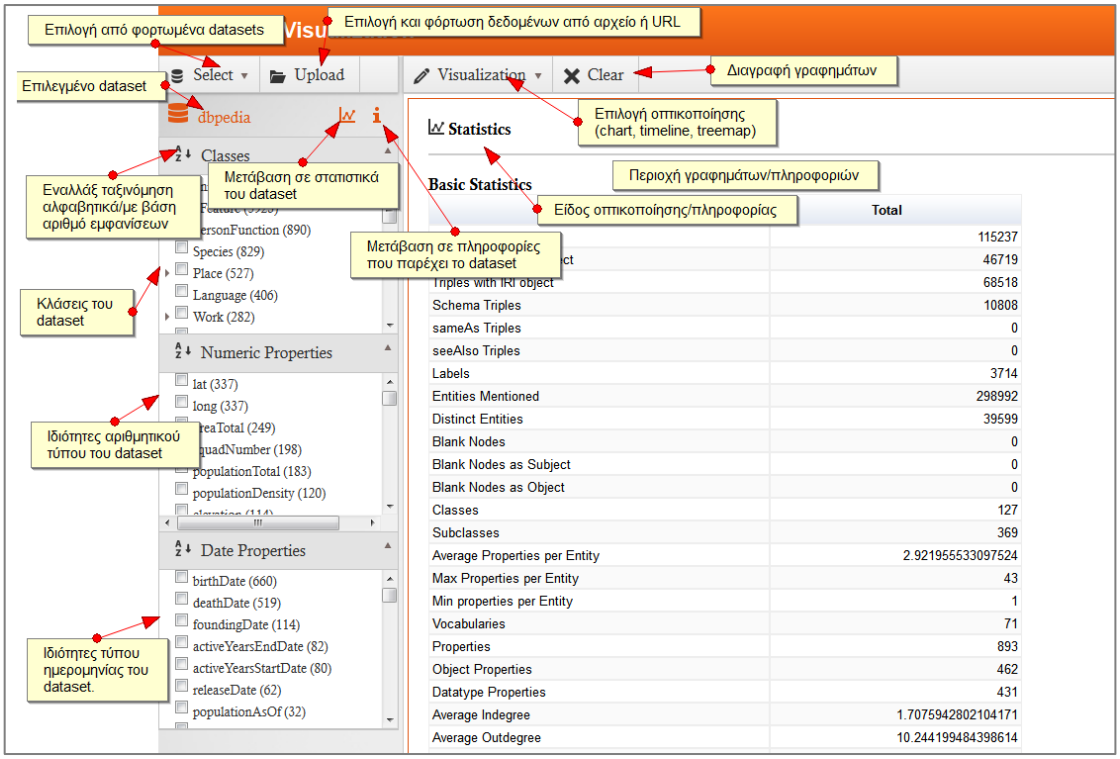
Στην ενότητα αυτή παρουσιάζουμε το σύστημα σύμφωνα με το παραπάνω σενάριο. Το κεφάλαιο αυτό, λειτουργεί και ως εγχειρίδιο χρήσης του προγράμματος.

7.2.1 Πρόσβαση στην εφαρμογή

Η πρόσβαση στην εφαρμογή γίνεται με την εισαγωγή της διεύθυνσης http://snf-73620.vm.okeanos.grnet.gr:8084/LinkedDataViz/ σε έναν web browser.

Αρχικά φορτώνεται στην εφαρμογή το dataset της DBpedia και εμφανίζονται τα στατιστικά του. Στην εικόνα βλέπουμε την πρώτη σελίδα.



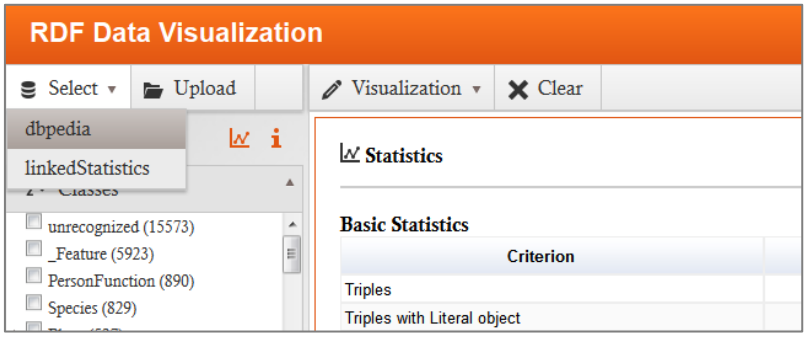


7.2.2 *Εισαγωγή δεδομένων*

Κατά την εισαγωγή δεδομένων φορτώνονται τα αντίστοιχα δεδομένα στα facets με τις κλάσεις και τις numeric και date properties της εφαρμογής. Εμφανίζονται επίσης τα στατιστικά που αφορούν το συνολικό σετ δεδομένων.

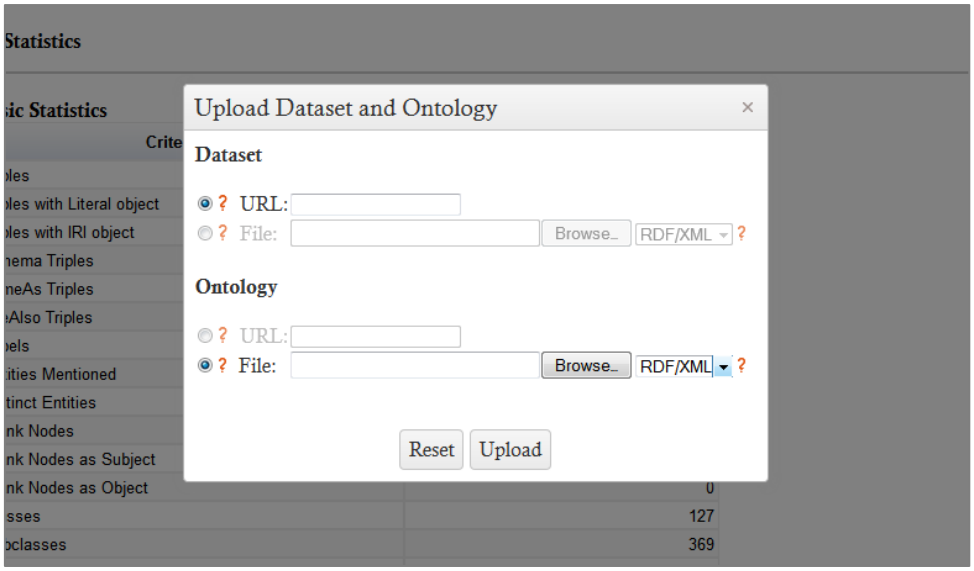
7.2.2.1 Εισαγωγή ήδη φορτωμένων σετ δεδομένων

Από το μενού “Select” επιλέγουμε ένα από τα διαθέσιμα datasets.



7.2.2.2 Εισαγωγή δεδομένων από αρχείο ή URL

Από το μενού επιλέγουμε “Upload” και εμφανίζεται το παρακάτω παράθυρο.



Στο παράθυρο αυτό επιλέγουμε το dataset που θέλουμε να οπτικοποιήσουμε και προαιρετικά οντολογία που περιγράφει το dataset.

Σε κάθε περίπτωση μπορούμε είτε να δώσουμε το URL ενός RDF dataset πχ. http://example.com/data.nt είτε να ανεβάσουμε ένα αρχείο από το τοπικό μας σύστημα. Στη δεύτερη περίπτωση επιλέγουμε από τη dropdown list την κωδικοποίηση του αρχείου.

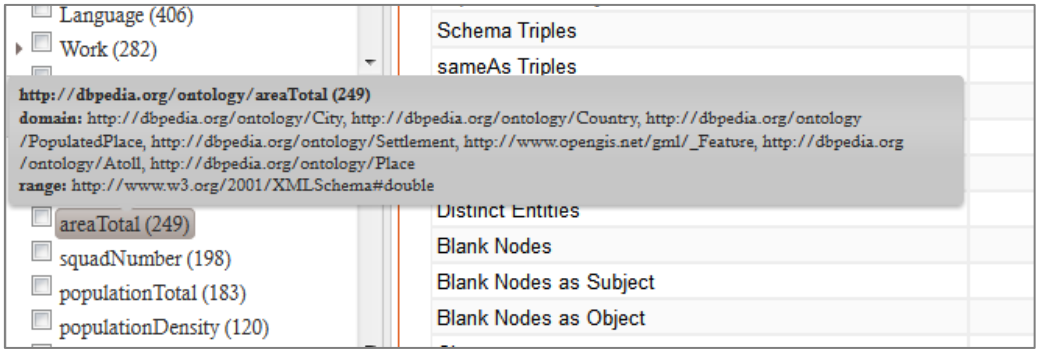
Πατώντας το πλήκτρο upload προχωράμε στην εισαγωγή, επεξεργασία και φόρτωση των δεδομένων μας.

7.2.3 Facets

Στο αριστερό τμήμα της οθόνης εμφανίζονται 3 panels που περιέχουν τα facets της εφαρμογής με τα οποία φιλτράρουμε το περιεχόμενο των διαγραμμάτων. Στο πρώτο εμφανίζονται οι κλάσεις της εφαρμογής μαζί με τον αριθμό instances τους, στο δεύτερο τα numeric properties με τον αριθμό εμφανίσεών τους και στο τρίτο τα date properties με τον αριθμό εμφανίσεών τους.

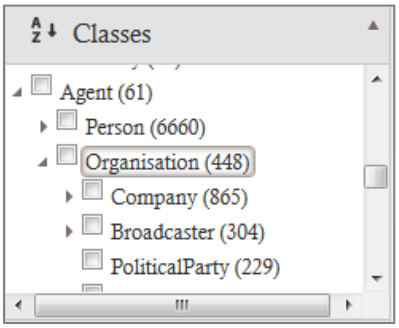
- Tooltips

  Σε κάθε στοιχείο των facets εμφανίζεται tooltip με ολόκληρο το URI του. Επιπλέον στα properties εμφανίζονται οι κλάσεις που έχουν ως domain και ως range.



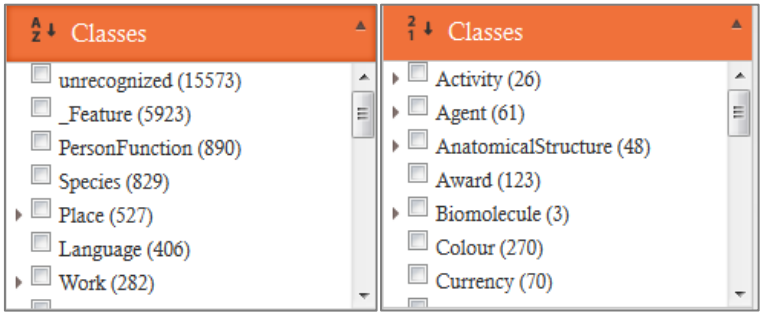
- Classes

  Οι κλάσεις της εφαρμογής παρουσιάζονται σε δενδρική μορφή. Αρχικά εμφανίζονται οι ανώτερες στην ιεραρχία κλάσεις. Κάνοντας κλικ στο βέλος αριστερά μιας κλάσης εμφανίζονται οι υποκλάσεις της.



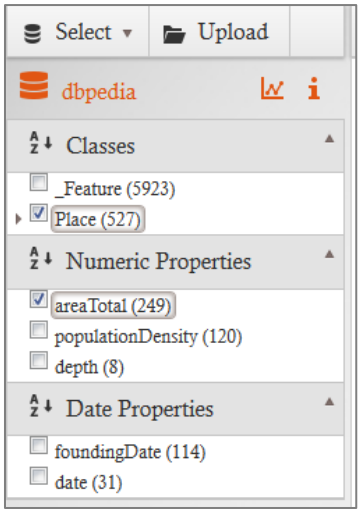
- Ταξινόμηση

  Τα στοιχεία των facets εμφανίζονται αρχικά ταξινομημένα σε φθίνουσα σειρά ως προς τον αριθμό εμφανίσεών τους/instances. Πατώντας το πλήκτρο στα αριστερά του τίτλου του panel τα περιεχόμενά του ταξινομούνται εναλλάξ σε αλφαβητική σειρά ή ως προς τον αριθμό εμφανίσεών τους.



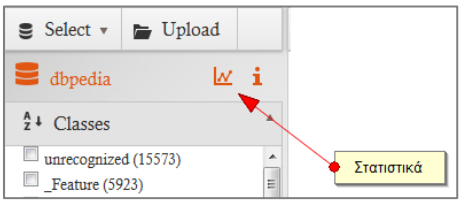
- Φιλτράρισμα

  Επιλέγοντας ένα στοιχείο από ένα panel (κάνοντας κλικ στο checkbox στα αριστερά του) τα υπόλοιπα facets φιλτράρονται αναλόγως. Συγκεκριμένα, όταν επιλέγεται μια κλάση, στα facets των numeric και date properties μένουν ορατά μόνο τα properties που έχουν την κλάση αυτή ως domain. Αντιστρόφως, όταν επιλεγεί ένα property (είτε numeric είτε date), στο facet των κλάσεων μένουν ορατές μόνο οι κλάσεις που έχει ως domain. Τα numeric και date facets δε σχετίζονται μεταξύ τους.



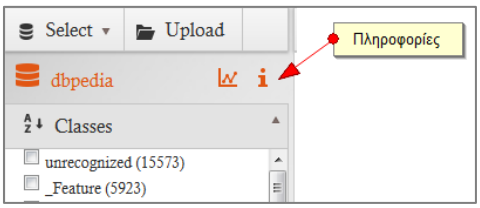
7.2.4 *Εμφάνιση Στατιστικών*

Πατώντας το πλήκτρο των στατιστικών εμφανίζονται στην κεντρική περιοχή της σελίδας πίνακες με στατιστικά στοιχεία για το επιλεγμένο dataset. Κάθε στήλη του πίνακα μπορεί να ταξινομηθεί σε αύξουσα ή φθίνουσα σειρά κάνοντας κλικ στον τίτλο της.

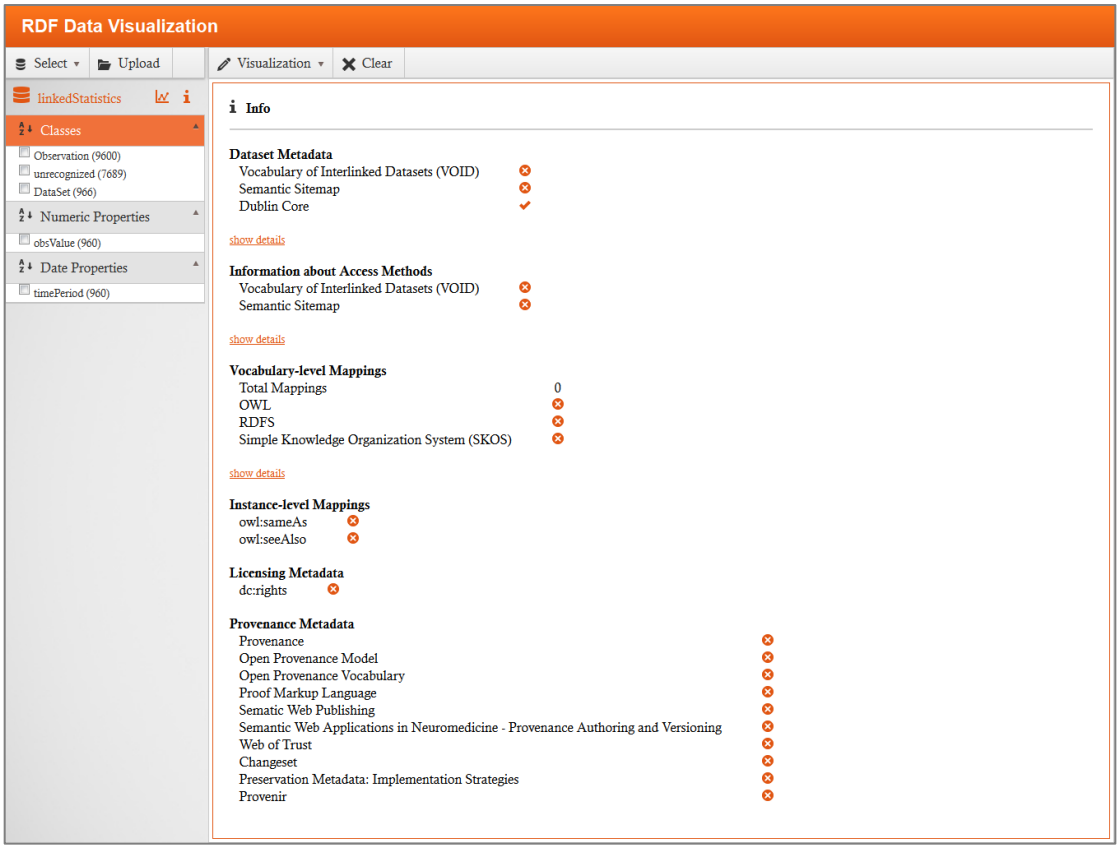


7.2.5 *Εμφάνιση Πληροφοριών*

Πατώντας το πλήκτρο των πληροφοριών εμφανίζονται στην κεντρική περιοχή της σελίδας πληροφορίες σχετικά με το επιλεγμένο dataset που προκύπτουν από τα μετα-δεδομένα που περιέχει.



Εμφανίζεται αντίστοιχο εικονίδιο όταν μια πληροφορία είναι διαθέσιμη ή όχι.



Πατώντας το πλήκτρο show details μπορούμε να δούμε αναλυτικά τις πληροφορίες που είναι διαθέσιμες σε κάθε κατηγορία.

Info

Dataset Metadata

Vocabulary of Interlinked Datasets (VOID)	✗
Semantic Sitemap	✗
Dublin Core	✓

hide details

Semantic Sitemap

sc:changefreq	✗
sc:dataDumpLocation	✗
sc:dataset	✗
sc:datasetLabel	✗
sc:datasetURI	✗
sc:lastmod	✗
sc:linkedDataPrefix	✗
sc:sampleURI	✗
sc:sparqlEndpointLocation	✗
sc:sparqlGraphName	✗

Dublin Core

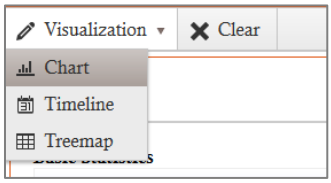
dcterms:available	✗
dcterms:contributor	✗
dcterms:created	✗
dcterms:creator	✗
dcterms:date	2012-06-05T02:17:49+0100
dcterms:dateAccepted	✗
dcterms:dateCopyrighted	✗
dcterms:dateSubmitted	✗
dcterms:description	✗
dcterms:modified	2012-06-05T02:17:49+0100
dcterms:publisher	✗
dcterms:source	http://epp.eurostat.ec.europa.eu/NavTree_prod/everybody/BulkDownloadListing?file=data%2Facf_p_ne.tsv.gz
dcterms:title	✗

Vocabulary of Interlinked Datasets (VOID)

void:dataDump	✗
void:Dataset	✗
void:exampleResource	✗
void:feature	✗
void:linkPredicate	✗
void:Linkset	✗
void:objectsTarget	✗
void:sparqlEndpoint	✗
void:subjectsTarget	✗
void:subset	✗
void:target	✗
void:TechnicalFeature	✗
void:uriLookupEndpoint	✗
void:uriRegexPattern	✗
void:vocabulary	✗

7.2.6 Διαγράμματα

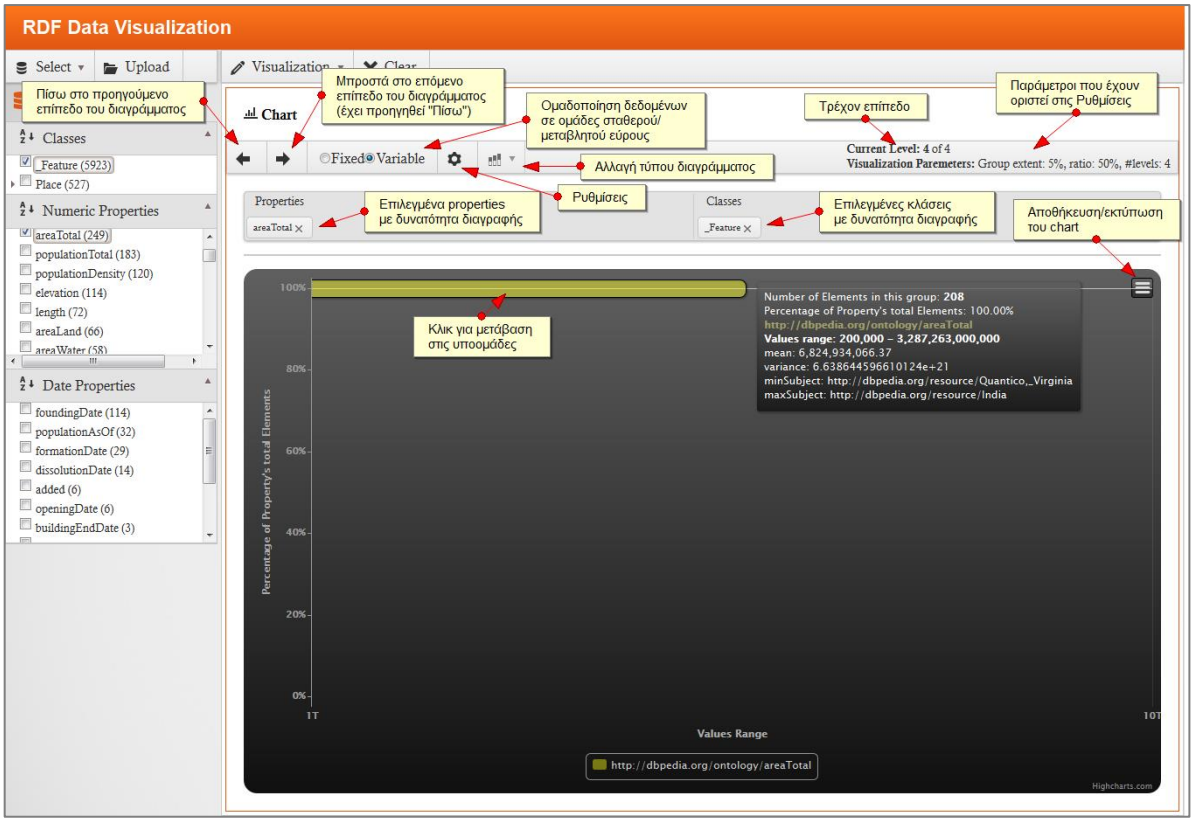
Από το μενού “Visualization” που βρίσκεται στο κεντρικό μέρος της σελίδας επιλέγουμε το είδος του διαγράμματος που επιθυμούμε να σχεδιαστεί. Για να διαγράψουμε όλα τα διαγράμματα πατάμε το “Clear” από το ίδιο μενού.



7.2.6.1 Chart

Για σχεδιασμό του Chart επιλέγουμε από το μενού “Visualization” το “Chart”. Τότε σχεδιάζεται chart που περιλαμβάνει τα properties που έχουν επιλεγεί στο facet Numeric Properties και παρουσιάζει το ανώτερο επίπεδο της ιεραρχίας στο οποίο έχουν οργανωθεί τα δεδομένα που ανήκουν στο κάθε property. Αν έχουν επιλεγεί και κλάσεις εμφανίζονται μόνο τα δεδομένα των properties που ανήκουν στις επιλεγμένες κλάσεις. Κάθε property σχεδιάζεται ως ξεχωριστή σειρά στο chart. Κάνοντας κλικ σε επιπλέον properties ή κλάσεις το chart επανασχεδιάζεται με τα νέα δεδομένα και εμφανίζεται πάλι το ανώτερο επίπεδο. Αφαιρώντας ένα property σβήνεται η σειρά που το απεικονίζει, χωρίς να επανασχεδιαστεί το διάγραμμα. Αφαιρώντας μία κλάση γίνεται επανασχεδιασμός του chart καθώς αλλάζουν για όλες τις σειρές τα δεδομένα που περιλαμβάνουν. Οι κλάσεις και τα properties που έχουν επιλεγεί εμφανίζονται πάνω από το chart με δυνατότητα αφαίρεσης.

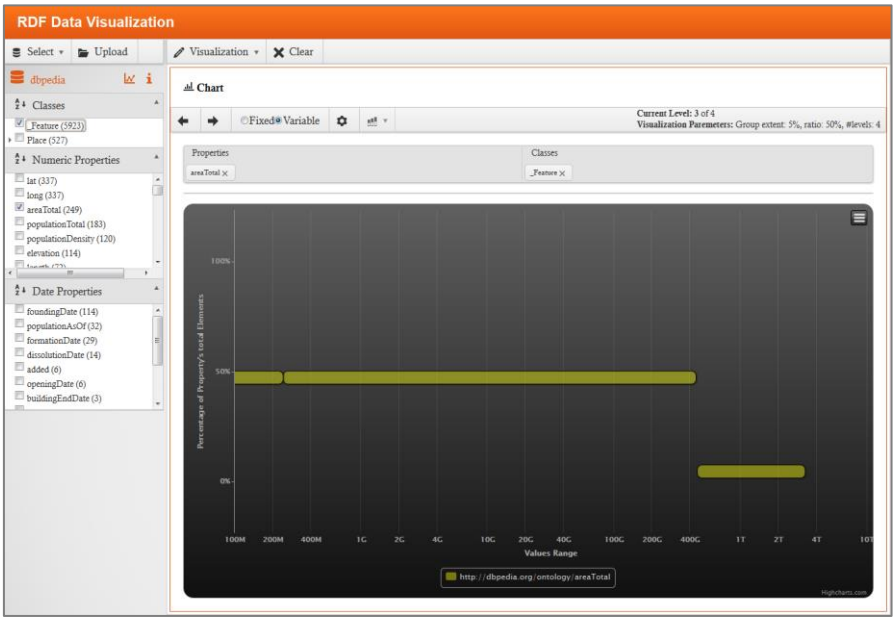
Στην εικόνα φαίνονται οι λειτουργίες που παρέχονται.

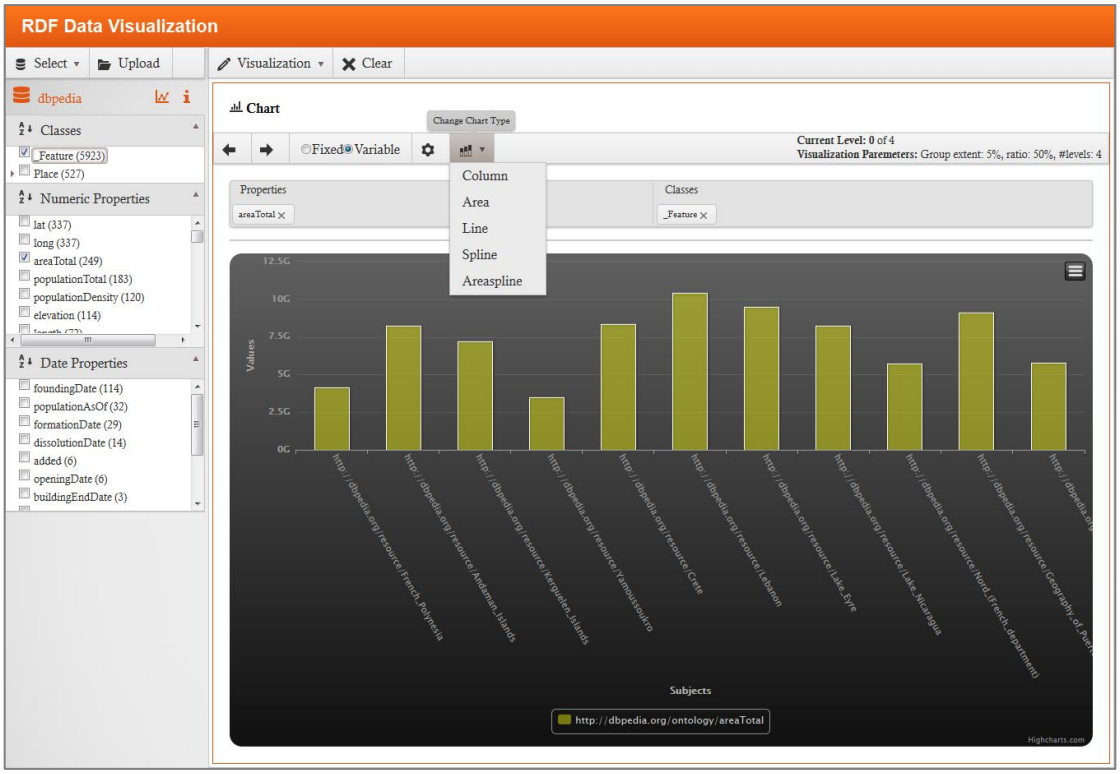


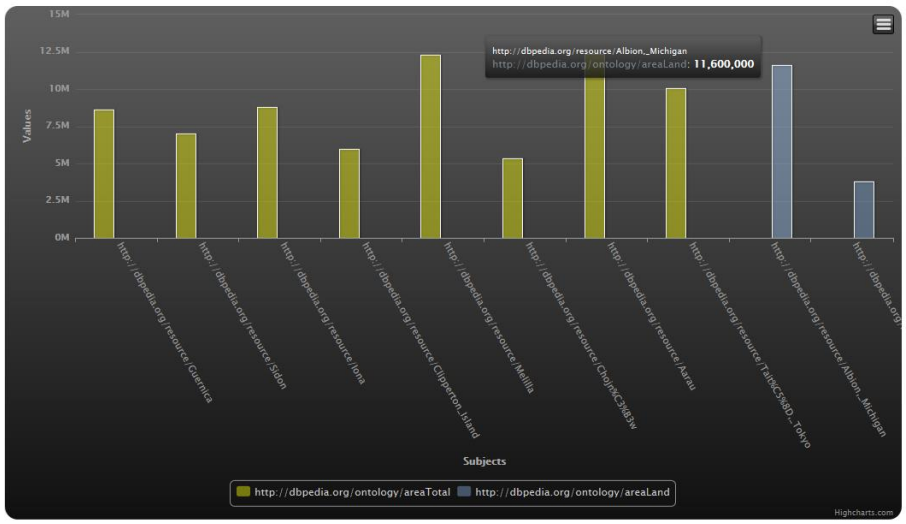
Αναλυτικότερα:

- Τα δεδομένα κάθε property χωρίζονται σε ομάδες οι οποίες ομαδοποιούνται περαιτέρω ώστε να σχηματίσουν μια ιεραρχία. Κάθε ομάδα έχει ένα εύρος τιμών και περιλαμβάνει δεδομένα των οποίων οι τιμές βρίσκονται μέσα στο εύρος αυτό. Στα ανώτερα επίπεδα της ιεραρχίας το εύρος είναι μεγαλύτερο. Όσο προχωράμε προς τα κάτω στην ιεραρχία το εύρος μικραίνει ώστε να επιλέξουμε τα δεδομένα που επιθυμούμε με μεγαλύτερη λεπτομέρεια. Στο τελευταίο επίπεδο της ιεραρχίας βλέπουμε τα ίδια τα δεδομένα που έχουν τιμή στο εύρος τιμών που επιλέξαμε.
- Κάθε property αντιπροσωπεύεται από μια ξεχωριστή σειρά διαφορετικού χρώματος.
- Κάθε μπάρα του chart αντιπροσωπεύει μια ομάδα (group) και εκτείνεται στο εύρος της ομάδας αυτής (x-άξονας). Στον y-άξονα φαίνεται το ποσοστό των δεδομένων που περιλαμβάνει η ομάδα του property σε σχέση με τα συνολικά δεδομένα του property.
- Σε κάθε ομάδα εμφανίζεται tooltip που περιλαμβάνει το συνολικό αριθμό δεδομένων της, το ποσοστό των δεδομένων που περιλαμβάνει η ομάδα του property σε σχέση με τα συνολικά δεδομένα του property, το εύρος τιμών της, το μέσο όρο και την τυπική απόκλιση των δεδομένων της, τα subjects με τη μικρότερη και τη μεγαλύτερη τιμή των δεδομένων της ομάδας (δείγματα δεδομένων που περιλαμβάνονται στην ομάδα) και τη σειρά-property στην οποία ανήκει η ομάδα. Το tooltip προτρέπει το χρήστη να κάνει κλικ στην αντίστοιχη μπάρα για να δει τα δεδομένα με τιμές στο εύρος τιμών που αντιπροσωπεύει η συγκεκριμένη μπάρα.

- Η πλοήγηση στο διάγραμμα γίνεται κάνοντας κλικ σε μια μπάρα-ομάδα. Ανάλογα με την τιμή των δεδομένων που επιθυμεί να δει ο χρήστης, μπορεί να επιλέξει την ομάδα αντίστοιχου εύρους τιμών. Τότε σχεδιάζεται νέο διάγραμμα με τις υποομάδες όλων των σειρών που βρίσκονται στο εύρος τιμών της ομάδας που επιλέξαμε. Οι υποομάδες που βρίσκονται εν μέρει στο εύρος αυτό "κόβονται" και εμφανίζεται μόνο το αποδεκτό τμήμα.
- Με τα πλήκτρα back-forward πάνω από το chart μπορούμε να μεταβούμε σε προηγούμενες και επόμενες καταστάσεις-επίπεδα της ιεραρχίας.
- Στο κατώτερο επίπεδο της ιεραρχίας παρουσιάζονται τα δεδομένα σε μορφή bar-chart. Στον x-άξονα παρουσιάζονται τα subjects και στο y-άξονα οι τιμές τους.







- Στο κατώτερο επίπεδο της ιεραρχίας μπορούμε να δούμε τα δεδομένα μας σε διαφορετικούς τύπους διαγραμμάτων:

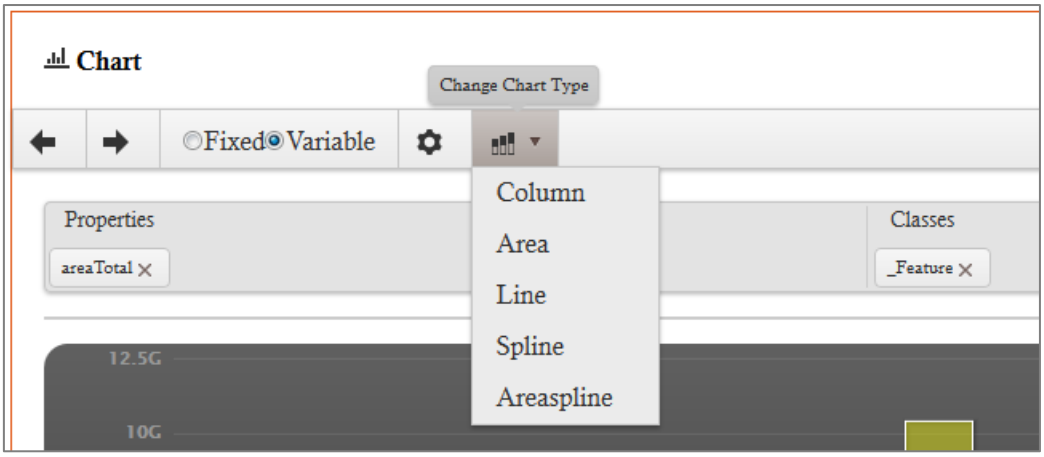
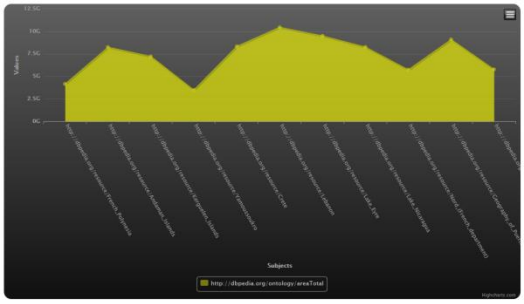
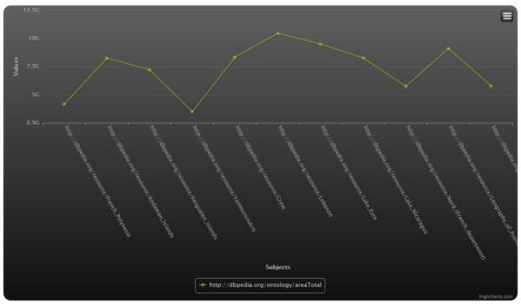
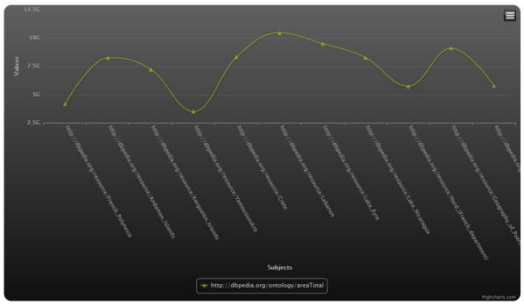
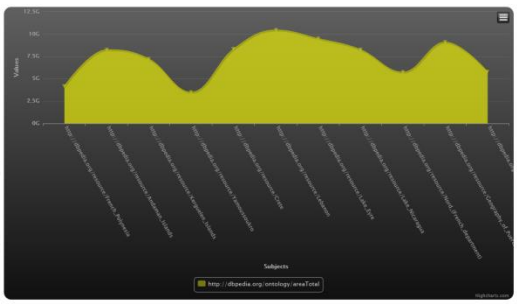


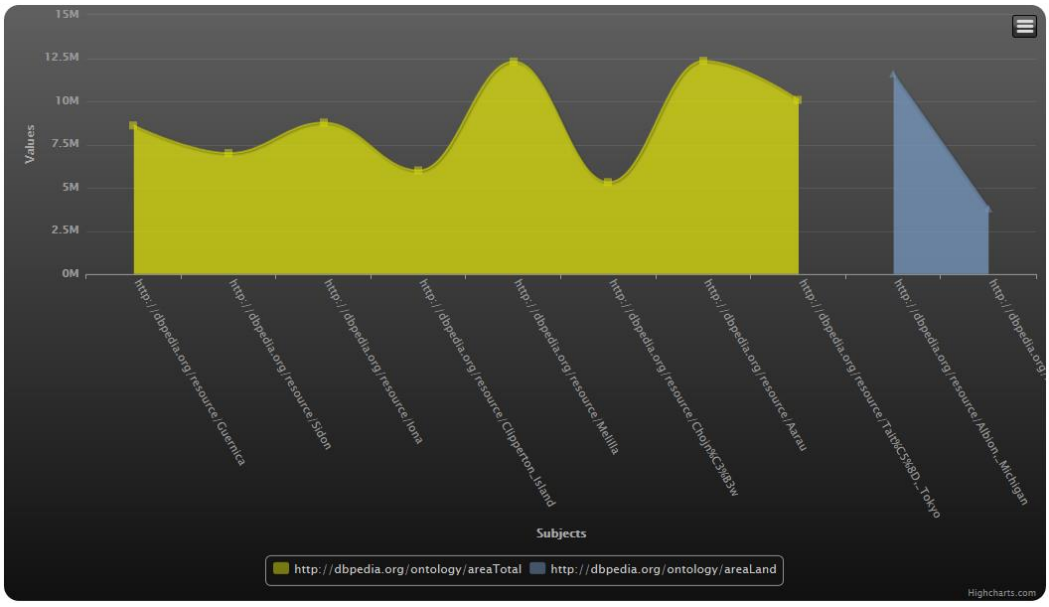
Chart
Change Chart Type
Fixed
Variable
Column
Area
Line
Spline
Areaspline
Properties
areaTotal
Classes
_Feature
12.5G
10G



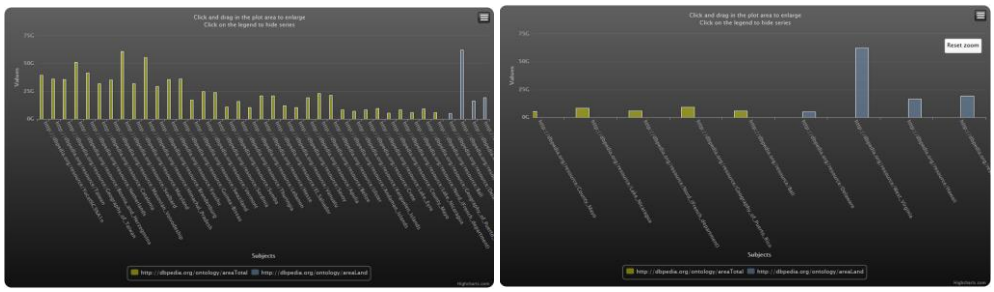




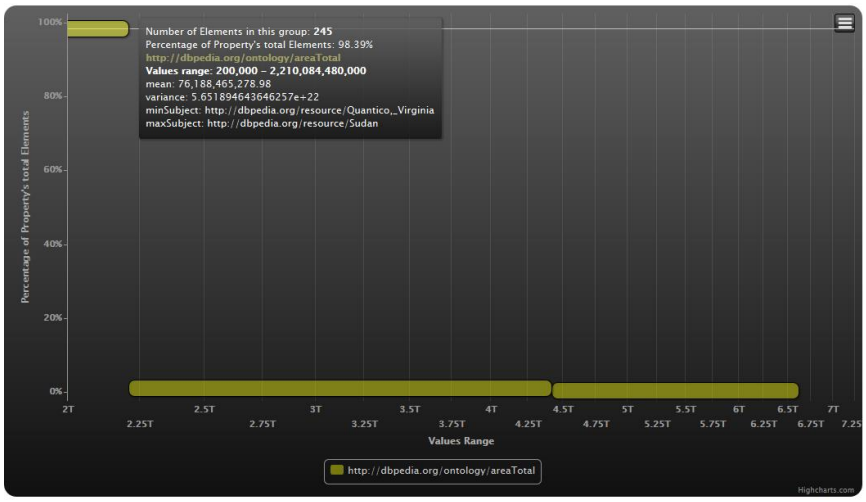


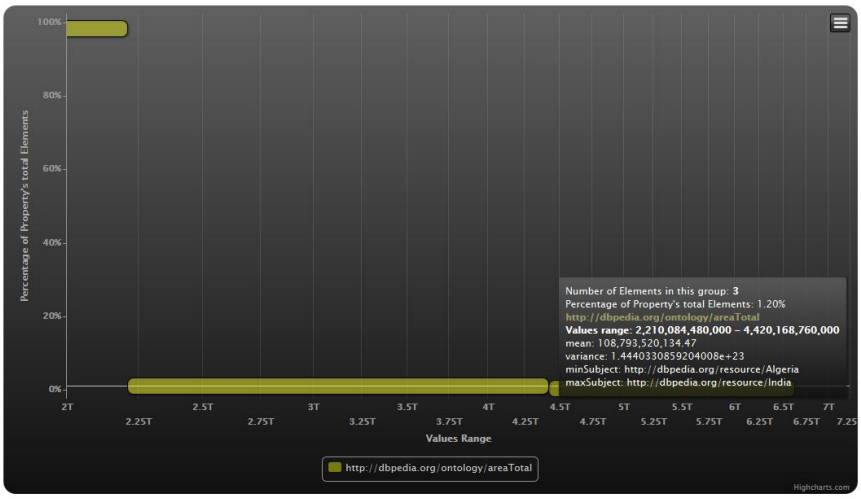

15M
12.5M
10M
7.5M
5M
2.5M
0M
Values
http://dbpedia.org/resource/Guernica
http://dbpedia.org/resource/Sidon
http://dbpedia.org/resource/Iona
http://dbpedia.org/resource/Clipperton_Island
http://dbpedia.org/resource/Melilla
http://dbpedia.org/resource/Chopin%C3%B3w
http://dbpedia.org/resource/Aarau
http://dbpedia.org/resource/Taito%C5%8D,_Tokyo
http://dbpedia.org/resource/Albion,_Michigan
Subjects
http://dbpedia.org/ontology/areaTotal
http://dbpedia.org/ontology/areaLand
Highcharts.com

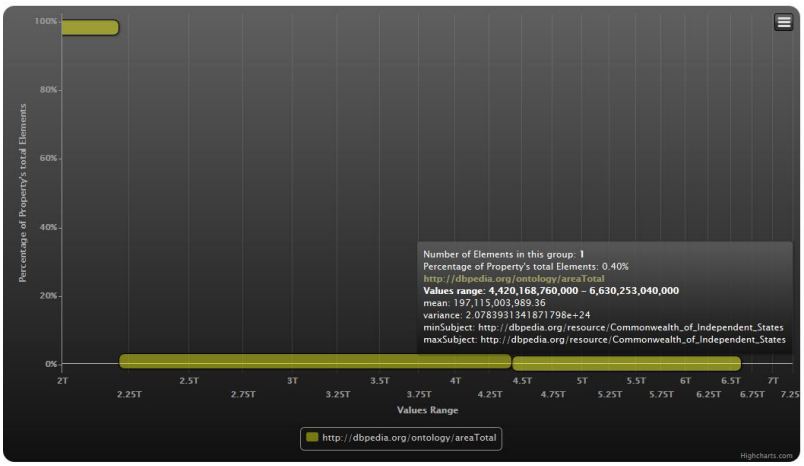
- Μπορούμε να μεγεθύνουμε μια περιοχή του διαγράμματος επιλέγοντάς την με το ποντίκι.



- Αρχικά η ομαδοποίηση των δεδομένων γίνεται σε ομάδες με σταθερό αριθμό στοιχείων. Επιλέγοντας “Fixed” από το μενού του chart τα δεδομένα ομοδοποιούνται σε ομάδες σταθερού εύρους τιμών και το chart επανασχεδιάζεται. Επιλέγοντας ξανά “Variable” το chart επανασχεδιάζεται με ομάδες σταθερού αριθμού στοιχείων.

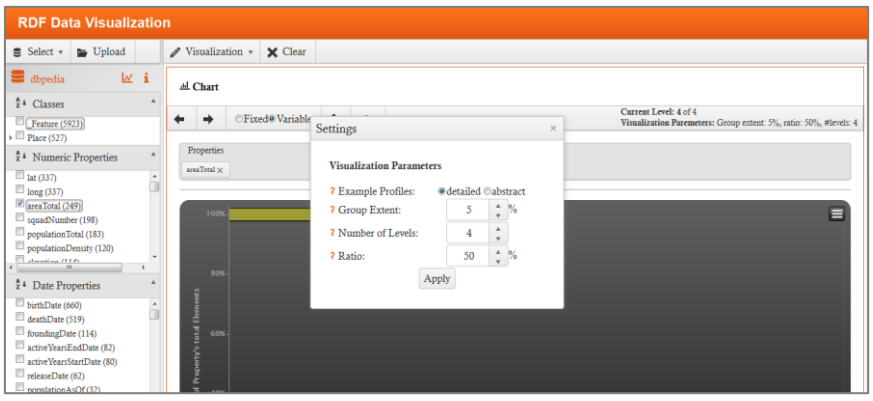


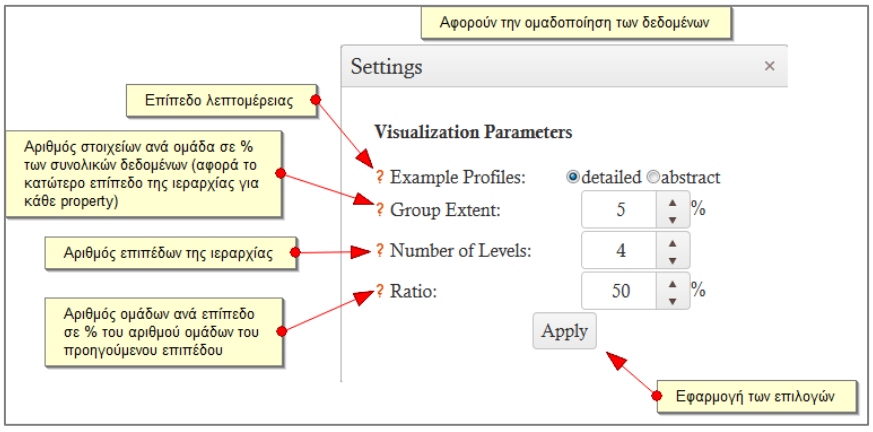




- Ρυθμίσεις

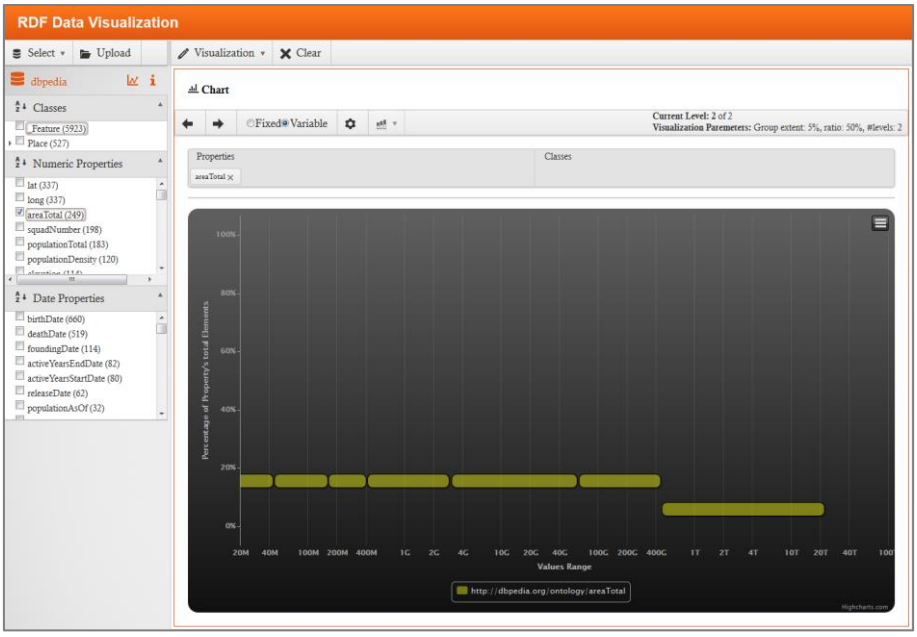
  Επιλέγοντας από το μενού του chart το πλήκτρο ρυθμίσεων εμφανίζεται ένα παράθυρο όπου μπορούμε να ρυθμίσουμε παραμέτρους της οπτικοποίησης.



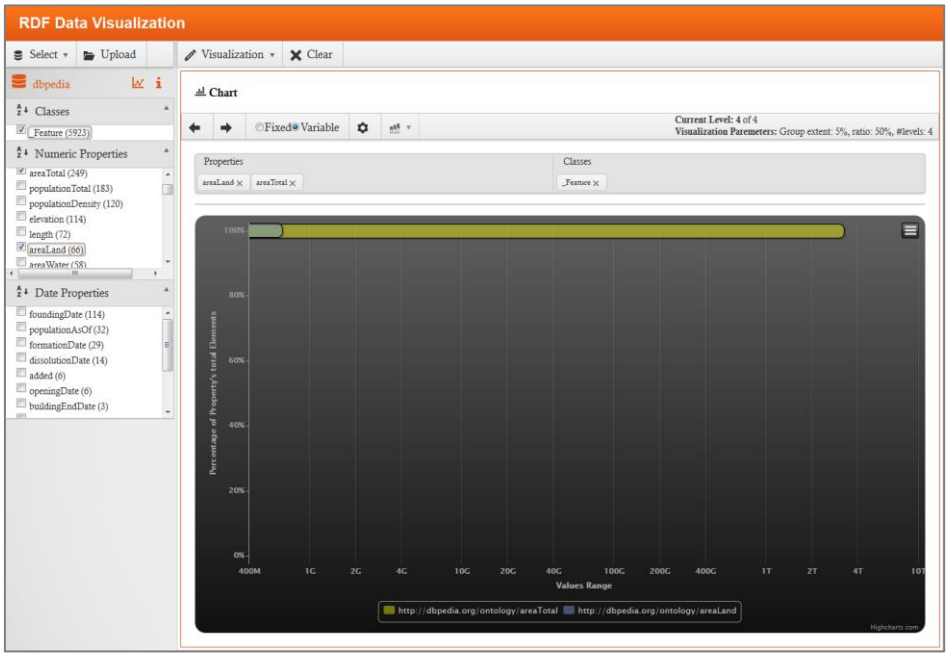


Επιλέγοντας ένα example profile μπορούμε να δούμε ενδεικτικές τιμές στους επόμενους δείκτες, τις οποίες μπορούμε να τροποποιήσουμε.

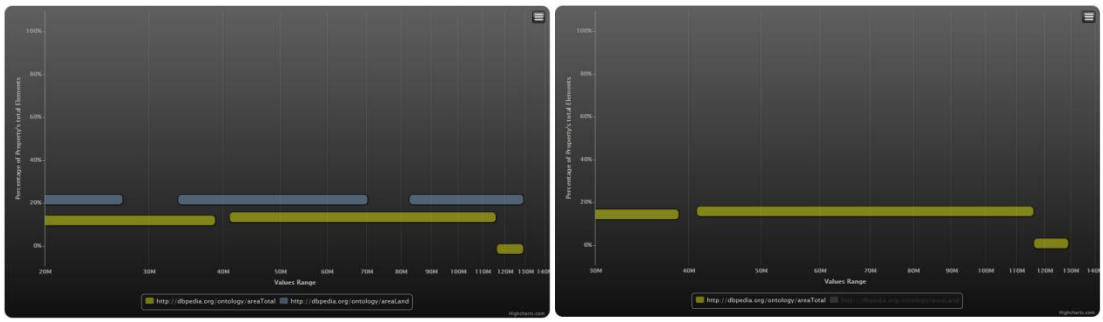
Με το πλήκτρο Apply εφαρμόζονται οι αλλαγές και το chart επανασχεδιάζεται.



- Επιλέγοντας μια επιπλέον ιδιότητα από το facet Numeric Properties το chart επανασχεδιάζεται ώστε να περιλαμβάνει και το νέο property.

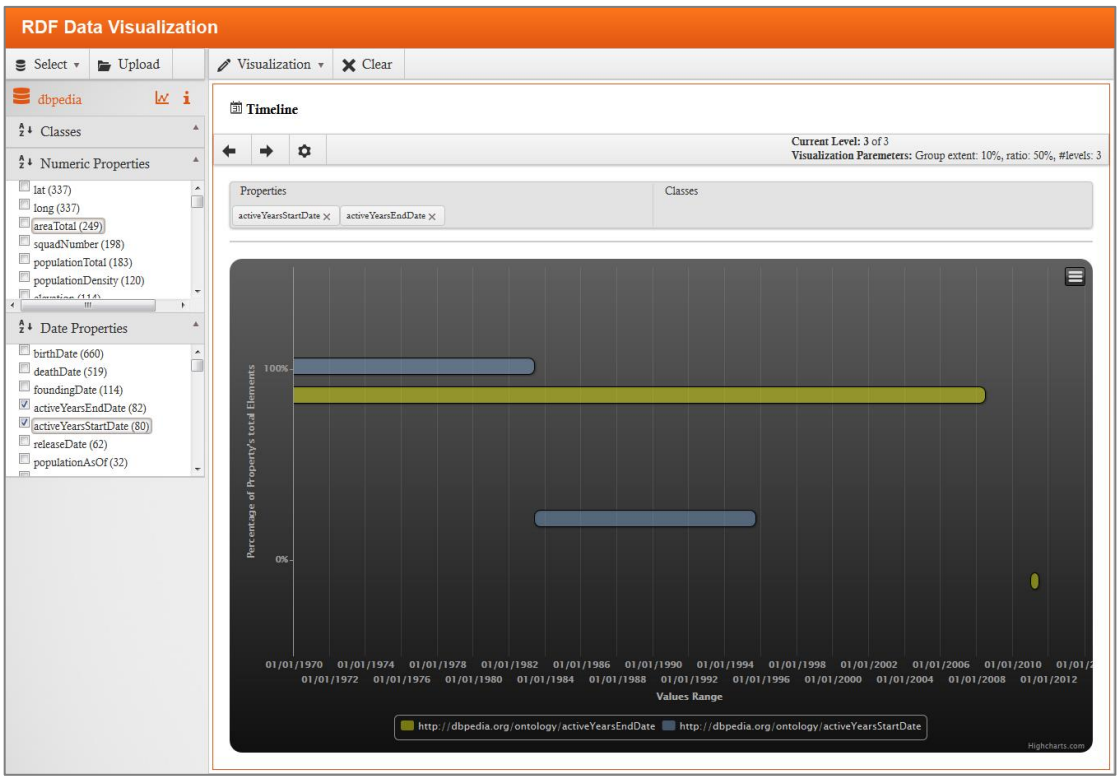


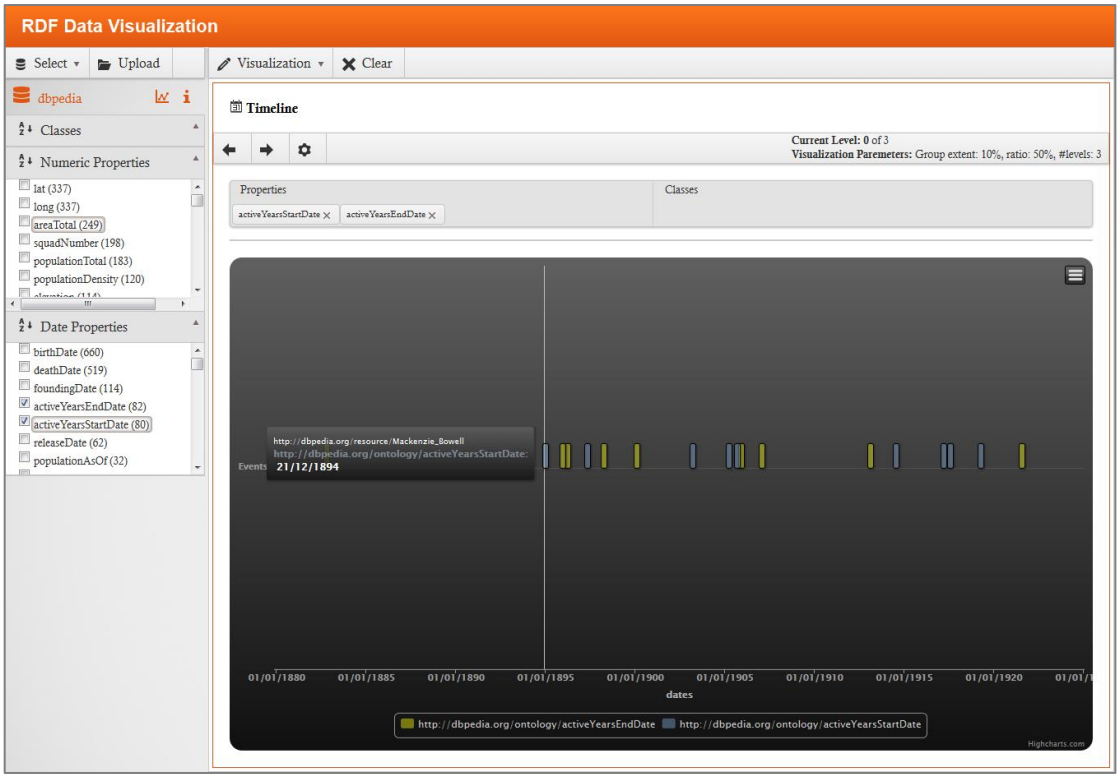
- Υπάρχει δυνατότητα να "κρύψουμε" μία ή περισσότερες σειρές από το chart χωρίς να τις σβήσουμε, με δυνατότητα επανεμφάνισης, για να δούμε καλύτερα τις υπόλοιπες. Για να γίνει αυτό κάνουμε κλικ στο όνομα της σειράς στο υπόμνημα, στο κάτω μέρος του chart.



7.2.6.2 Timeline

Για σχεδιασμό του Timeline επιλέγουμε από το μενού “Visualization” το “Timeline”. Τα διαγράμματα είναι όμοια με το Chart με εξαίρεση το κατώτερο επίπεδο όπου παρουσιάζονται τα τελικά δεδομένα σε μορφή Timeline. Τα properties που οπτικοποιούνται επιλέγονται από το facet Date Properties και τα δεδομένα τους μπορούν να φιλτραριστούν από τις κλάσεις.

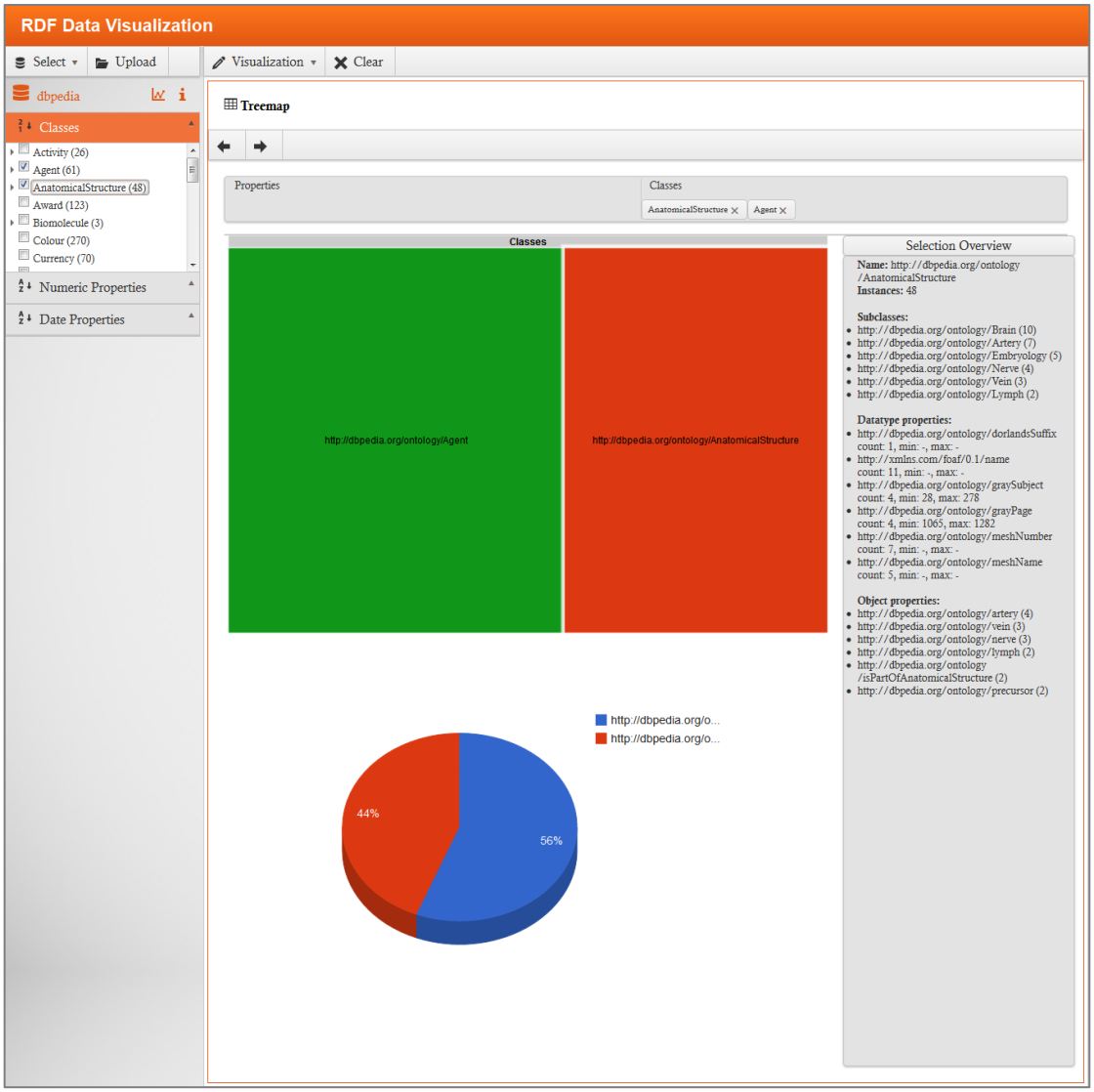




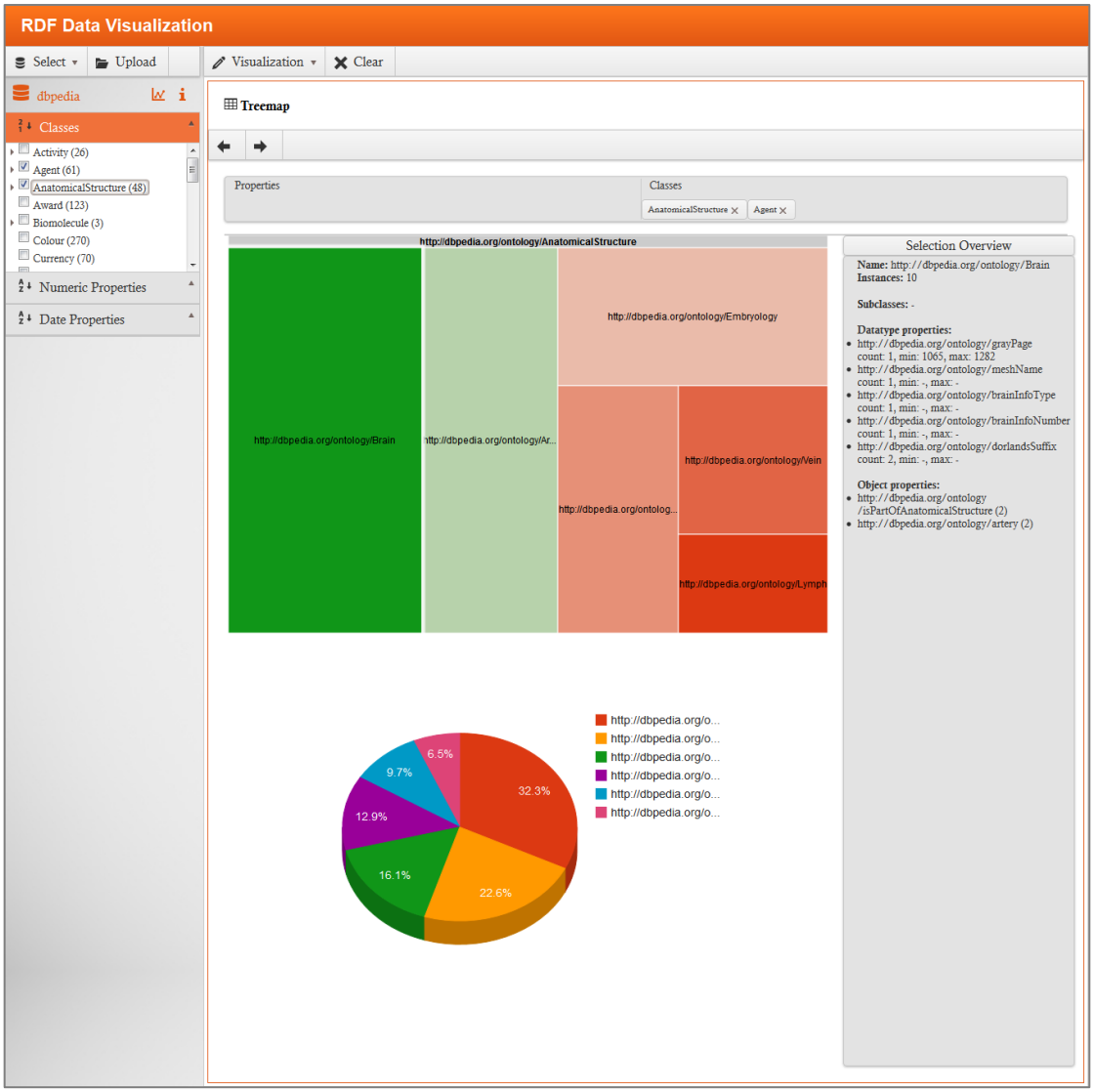
7.2.6.3 Treemap

Για σχεδιασμό του Treemap επιλέγουμε από το μενού “Visualization” το “Treemap”. Τότε σχεδιάζεται Treemap που περιλαμβάνει τις κλάσεις που έχουν επιλεγεί στο facet Classes μαζί με Pie Chart. Το μέγεθος κάθε τετραγώνου στο treemap αντιπροσωπεύει τον αριθμό instances της κλάσης που απεικονίζει ενώ το Pie Chart δείχνει το ποσοστό που κατέχει κάθε κλάση σε σχέση με τις υπόλοιπες που απεικονίζονται. Αν έχουν επιλεγεί και υποκλάσεις μιας κλάσης, συμπεριλαμβάνεται μόνο η ανώτερη στην ιεραρχία των επιλεγμένων κλάσεων. Αν προστεθεί ή αφαιρεθεί μια κλάση το διάγραμμα επανασχεδιάζεται. Οι κλάσεις και τα properties που έχουν επιλεγεί εμφανίζονται πάνω από το chart με δυνατότητα αφαίρεσης.

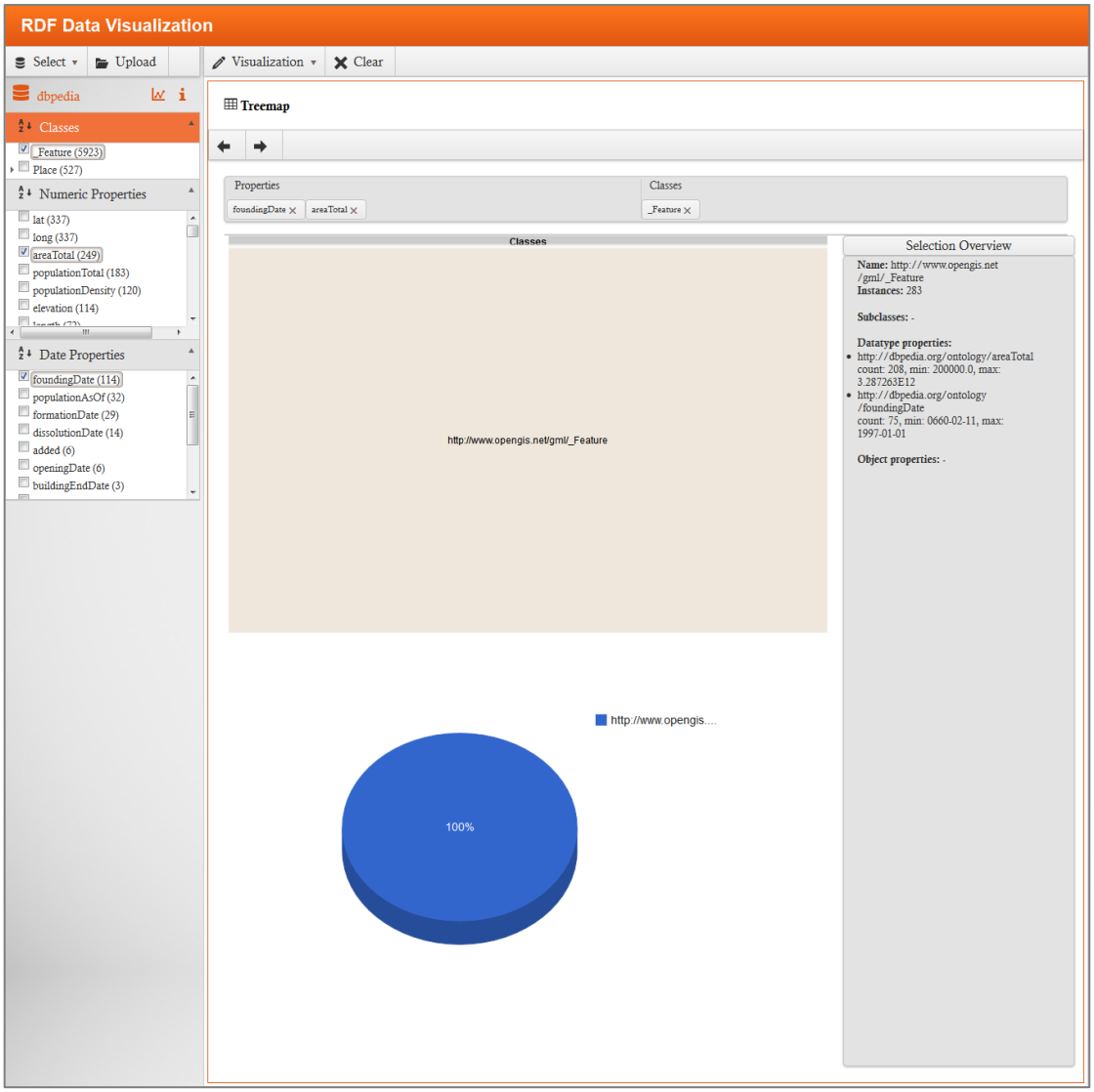
- Πηγαίνοντας με το ποντίκι πάνω σε μια κλάση εμφανίζονται στα δεξιά του διαγράμματος πληροφορίες και στατιστικά στοιχεία για την κλάση αυτή. Συγκεκριμένα εμφανίζεται ο αριθμός instances της κλάσης, οι υποκλάσεις της με τον αντίστοιχο αριθμό instances, οι datatype properties με τον αντίστοιχο αριθμό εμφανίσεών τους και τη μέγιστη και ελάχιστη τιμή τους και οι object properties με τον αντίστοιχο αριθμό εμφανίσεών τους. Οι datatype και object properties αφορούν properties σε τριπλέτες των οποίων το subject ή το object ανήκει στην κλάση.



- Κάνοντας κλικ σε μία κλάση εμφανίζονται οι υποκλάσεις της μαζί με τις ανάλογες πληροφορίες.



- Αν επιλεγούν properties φιλτράρονται τα instances της κλάσης και στον υπολογισμό των instances κλάσεων και properties λαμβάνονται υπόψη μόνο οι τριπλέτες με τις επιλεγμένες properties.


RDF Data Visualization
Select
Upload
Visualization
Clear
dbpedia
Classes
_Feature (5923)
Place (527)
Numeric Properties
lat (337)
long (337)
areaTotal (249)
populationTotal (183)
populationDensity (120)
elevation (114)
Date Properties
foundingDate (114)
populationAsOf (32)
formationDate (29)
dissolutionDate (14)
added (6)
openingDate (6)
buildingEndDate (3)
Treemap
Properties
foundingDate
areaTotal
Classes
_Feature
Classes
http://www.opengis.net/gml/_Feature
Selection Overview
Name: http://www.opengis.net/gml/_Feature
Instances: 283
Subclasses: -
Datatype properties:
http://dbpedia.org/ontology/areaTotal
count: 208, min: 200000.0, max: 3.287263E12
http://dbpedia.org/ontology/foundingDate
count: 75, min: 0660-02-11, max: 1997-01-01
Object properties: -
http://www.opengis....
100%

8

Επίλογος

8.1 Σύνοψη

Στη διπλωματική αυτή ασχοληθήκαμε με την οπτικοποίηση διασυνδεδεμένων δεδομένων. Ο ολοένα αυξανόμενος όγκος των διασυνδεδεμένων δεδομένων κάνει επιτακτική την ανάγκη ανάπτυξης γενικών εργαλείων οπτικοποίησής τους που επιτρέπουν τον αποδοτικό χειρισμό και την παρουσίαση μεγάλου όγκου ετερογενών δεδομένων, απευθυνόμενα σε απλούς χρήστες εκτός από γνώστες των αντίστοιχων τεχνολογιών.

Στα πλαίσια της διπλωματικής μελετήσαμε αρχικά διάφορες τεχνικές οπτικής αναπαράστασης δεδομένων που υποστηρίζονται από JavaScript βιβλιοθήκες για την οπτικοποίηση δεδομένων στον ιστό. Επιλέξαμε τη βιβλιοθήκη Highcharts για charts και timeline, και τη βιβλιοθήκη Google Charts για treemaps και pie.

Στη συνέχεια σχεδιάσαμε το ενιαίο μοντέλο αναπαράστασης ενός Linked Dataset και το μηχανισμό συλλογής στατιστικών από αυτό. Το μοντέλο αυτό είναι ανεξάρτητο από την εκάστοτε τεχνική και βιβλιοθήκη οπτικοποίησης και μπορεί να αξιοποιηθεί για περισσότερους του ενός τρόπους οπτικοποίησης ενός dataset. Παράλληλα ομαδοποιεί τα δεδομένα και τα οργανώνει σε ιεραρχίες.

Αναπτύχθηκε ακόμα μια web-based εφαρμογή που αξιοποιεί το μοντέλο αυτό για οπτικοποίηση σε συνδυασμό με παρουσίαση στατιστικών, πλοήγηση και επιλογή φίλτρων με

γραφικό τρόπο (faceted search) στο set δεδομένων. Η εφαρμογή μπορεί να χρησιμοποιηθεί για οποιδήποτε dataset ανεξαρτήτως του τομέα στον οποίο ανήκει και χειρίζεται αποδοτικά μεγάλα datasets παρέχοντας δυνατότητα για overview των δεδομένων και zoom in στις περιοχές ενδιαφέροντος του χρήστη. Απευθύνεται τόσο σε απλούς χρήστες όσο και σε γνώστες των τεχνολογιών σημασιολογικού Ιστού παρέχοντας ένα εύχρηστο User Interface που καθοδηγεί το χρήστη. Τέλος έγινε αξιολόγηση και έλεγχος της εφαρμογής μέσω σεναρίων λειτουργίας.

8.2 Μελλοντικές επεκτάσεις

Παρακάτω προτείνουμε πιθανές επεκτάσεις της εφαρμογής:

- Σύνδεση της εφαρμογής με SPARQL endpoint και οπτικοποίηση αποτελέσματος ερωτημάτων.
- Δυνατότητα εκτέλεσης ερωτημάτων σε φορτωμένο dataset και οπτικοποίηση αποτελέσματος.
- Δυνατότητα αποθήκευσης του dataset που έχει ανεβάσει ο χρήστης μαζί με το μοντέλο οπτικοποίησης και στατιστικών που δημιουργεί για αυτό η εφαρμογή.
- Παροχή περισσότερων οπτικοποιήσεων όπως map, graph κλπ. που επιτρέπουν στο χρήστη να εξερευνήσει περισσότερες πτυχές του dataset.
- Βελτίωση των οπτικοποιήσεων για παροχή περισσότερων επιλογών για overview των δεδομένων και zoom-in.
- Βελτίωση του μοντέλου και της υλοποίησης για καλύτερη απόδοση και ταχύτητα κατά την προεπεξεργασία δεδομένων.

9

Βιβλιογραφία

1. Dadzie, Aba-Sah and Rowe, Matthew. (2011), Approaches to visualising Linked Data: A survey, *Semantic Web – Interoperability, Usability, Applicability*, IOS Press, Vol. 2, pp. 89-124.

2. Protégé. [Online] http://protege.stanford.edu/.

3. RDF Gravity. [Online] http://semweb.salzburgresearch.at/apps/rdf-gravity/.

4. Welkin. [Online] http://simile.mit.edu/welkin/.

5. Freebase. [Online] http://www.freebase.com/.

6. Pubby. [Online] http://wifo5-03.informatik.uni-mannheim.de/pubby/.

7. Disco. [Online] http://wifo5-03.informatik.uni-mannheim.de/bizer/ng4j/disco/.

8. Berners-Lee, Tim, et al., et al. (2006), Tabulator: Exploring and analyzing linked data on the semantic web., *3rd International Semantic Web User Interaction.*

9. de Araújo, Samur F. C. and Schwabe, Daniel. (2009), Explorator: a tool for exploring RDF data through direct., *Linked Data on the Web*.

10. Brunetti, Josep Maria, Gil, Rosa and García, Roberto (2012), Facets and Pivoting for Flexible and Usable, *Interacting with Linked Data*, Heraklion, Greece.

11. Hastrup, Tuukka, Cyganiak, Richard and Bojars, Uldis. (2008), Browsing Linked Data with Fenfire, *Linked Data on the Web*.

12. RDF-Gravity. [Online] http://semweb.salzburgresearch.at/apps/rdf-gravity/.

13. IsaViz. [Online] http://www.w3.org/2001/11/IsaViz/.

14. Karger, David and M.C., Schraefel (2006), The Pathetic Fallacy of RDF, *International Workshop on the Semantic Web and User Interaction*, Athens, Georgia

15. Skjaeveland, Martin G. (2012), Sgvizler: A JavaScript Wrapper for Easy Visualization of SPARQL Result Sets, *Extended Semantic Web Conference*.

16. Stuhr, Magnus, Roman, Dumitru and Norheim, David. Bonn, Germany (2011), LODWheel – JavaScript-based Visualization of RDF Data., *International Semantic Web Conference*.

17. Brunetti, Josep Maria, Auer, Sören and García, Roberto (2012), The Linked Data Visualization Model.

18. LinkedGeoData. [Online] http://linkedgeodata.org/LGD%20Browser.

19. map4rdf. [Online] http://oeg-dev.dia.fi.upm.es/map4rdf/.

20. Schneiderman, Ben. (1996), The eyes have it: a task by data type taxonomy. *IEEE Symposium on Visual Languages.* pp. 336-343.

21. Elmqvist, Niklas and Fekete, Jean-Daniel. (2009), Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 16, pp. 439 - 454 .

22. Schneiderman, Ben. (1992), Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics (TOG)*, Vol. 11, pp. 92-99.

23. Wang, Taowei David and Parsia, Bijan. (2006). CropCircles: Topology Sensitive Visualization of OWL Class Hierarchies. *Proceedings of the 5th International Semantic Web Conference*. pp. 695-708.

24. Plaisant, Catherine, Grosjean, Jesse and Bederson, Benjamin B. (2002). SpaceTree: supporting exploration in large node link tree, design evolution and empirical evaluation. *IEEE Symposium on Information Visualization.* pp. 57-64.

25. Auer, Sören, et al., (2012)., LODStats - An Extensible Framework for High-performance Dataset Analytics. *Proceedings of the 18th international conference on Knowledge Engineering and Knowledge Management.* pp. 353-362.

26. make-void. [Online] https://github.com/cygri/make-void.

27. Apache Jena Framework. [Online] http://jena.apache.org/.

28. VOID. [Online] 2013. http://www.w3.org/TR/void/.

29. SPARQL. [Online] http://www.w3.org/TR/rdf-sparql-query/.

30. Langegger, A. and Woss, W. (2009). RDFStats - An Extensible RDF Statistics Generator and Library. *20th International Workshop on Database and Expert Systems Application*. pp. 79-83.

31. Linked Data Tools. [Online] http://www.linkeddatatools.com/semantic-web-basics.

32. Wikipedia. [Online] https://en.wikipedia.org/wiki/Semantic_web.

33. Wikipedia. [Online] https://en.wikipedia.org/wiki/Linked_data.

34. Linked Data. [Online] http://linkeddata.org/.

35. LODCloud. [Online] http://lod-cloud.net/state/.

36. Wikipedia. [Online] https://en.wikipedia.org/wiki/Linked_data.

37. W3C. [Online] http://www.w3.org/TR/rdf-schema/.

38. W3C. [Online] http://www.w3.org/TR/owl-ref/.

39. W3. [Online] http://www.w3.org/TR/rdf-sparql-query/.

40. Wikipedia. [Online] http://en.wikipedia.org/wiki/Named_graph.

41. Zaveri, Amrapali, Rula Anisa, Maurino, Andrea, Pietrobon, Ricardo, Lehmann, Jens and Auer, Soren. (2012), Quality Assessment Methodologies for Linked Open Data, *Semantic Web – Interoperability, Usability, Applicability*, IOS Press.