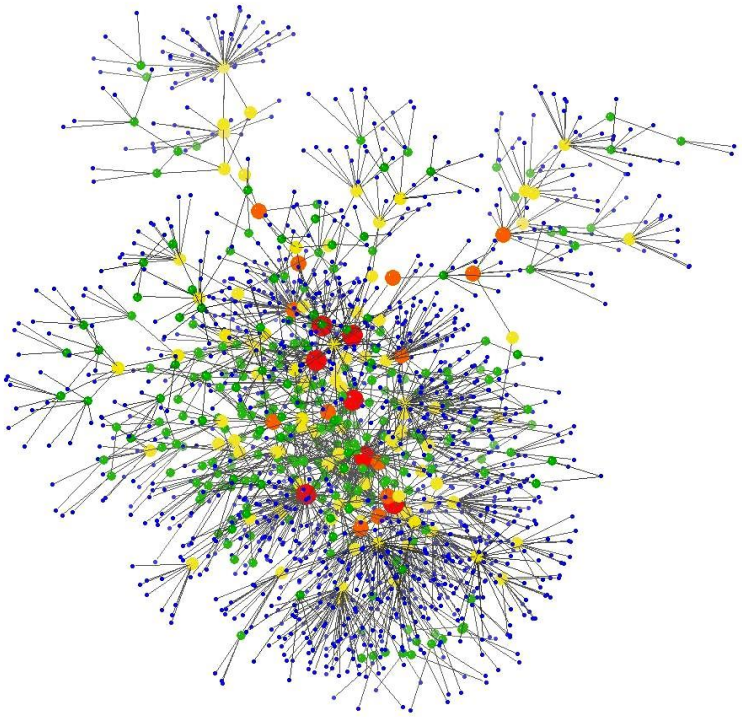
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
UNIVERSITY OF PIRAEUS

2014

Εργαλείο Οπτικοποίησης Κατατμημένων Οντολογιών

Διπλωματική εργασία με θέμα εργαλείο οπτικοποίησης κατατμημένων οντολογιών



Επιβλέπων Καθηγητής: Βούρος Γεώργιος

Ψύχας Αλέξανδρος
ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ ΗΛΕΚΤΡΟΝΙΚΗΣ ΜΑΘΗΣΗΣ
ΤΜΗΜΑΤΟΣ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ ΠΑΝΕΠΙΣΤΗΜΙΟΥ ΠΕΙΡΑΙΩΣ
2011-2013

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον καθηγητή Βούρο Γεώργιο και

το διδάκτορα Σαντιπαντάκη Γεώργιο,

για τη σημαντικότατη βοήθεια και καθοδήγηση τους.

Περιεχόμενα

1. Εισαγωγή

Ο σημασιολογικός ιστός (Semantic Web) αποτελείται από μια ποικιλία αναπτυσσόμενων τεχνολογιών του παγκόσμιου ιστού (World Wide Web (WWW)) . Οι χρήσεις του και οι δυνατότητες του ποικίλουν: Από τη χρήση του σε μηχανές αναζήτησης, την ιεράρχηση και κατηγοριοποίηση των ψηφιακών δεδομένων, μέχρι και τη χρήση του από προσωπικούς πράκτορες (Personal Agents) για την ευφυή επεξεργασία της πληροφορίας. Σκοπός του σημασιολογικού ιστού είναι η αναπαράσταση και η περιγραφή των δεδομένων με τέτοιο τρόπο, ώστε να είναι κατανοητά και να μπορούν να επεξεργαστούν από υπολογιστικά συστήματα. Ο τρόπος που περιγράφονται τα δεδομένα στα πλαίσια του σημασιολογικού ιστού για να επιτευχθεί αυτός ο σκοπός, είναι κυρίως μέσω της αξιοποίησης λεξικών και οντολογιών [1].

Οι οντολογίες αποτελούν έναν επίσημο τρόπο περιγραφής δεδομένων που αφορούν οντότητες του πραγματικού κόσμου. Η κυρίαρχη γλώσσα για τον καθορισμό των οντολογιών είναι η OWL (Web Ontology Language) [13]. Η γλώσσα OWL δημιουργήθηκε το 2002, σε μια προσπάθεια δημιουργίας μια επίσημης γλώσσας με τυποποιημένο συντακτικό, σημασιολογία και υψηλή εκφραστικότητα, για τη περιγραφή των δεδομένων. Το 2008 εισήχθη η ανανεωμένη και βελτιωμένη έκδοση της OWL, η γλώσσα OWL2[11]. Τα κύρια καινούρια χαρακτηριστικά της OWL2 είναι η «προσαρμογή» της εκφραστικότητας της γλώσσας σε διαφόρους τρόπους χρήσης της και η επέκταση των τύπων δεδομένων.

Η δημιουργία των οντολογιών πυροδότησε την δημιουργία γλωσσών για τη σύνδεση των οντολογιών. Οι γλώσσες συσχέτισης οντολογιών έχουν στόχο την σύνδεση οντολογιών μεταξύ τους. Υπάρχουν ποικίλες γλώσσες για την υλοποίηση αυτών των συνδέσεων. Παραδοσιακά η συσχέτιση μεταξύ οντολογιών γίνεται μέσω της σύνδεσης των εννοιών που περιέχουν, πιο συγκεκριμένα τη σύνδεση των κλάσεων των οντολογιών μέσω αντιστοιχίσεών τους. Μια γλώσσα που ορίζει τη σύνδεση κλάσεων μεταξύ οντολογιών είναι η γλώσσα C-OWL(Contextualizing-OWL)[3], όπου είναι και η γλώσσα μαζί με τις επεκτάσεις της οποίας, βασίζεται η παρούσα διπλωματικής. Άλλες γλώσσες που επισημοποιούν την συσχέτιση οντολογιών είναι η Expressive and Declarative Ontology Alignment Language (EDOAL)[9] και τα «Correspondence Patterns for ontology»[10].

1.1. Περίληψη

Η παρούσα διπλωματική εργασία έχει ως σκοπό τη δημιουργία ενός συστήματος για την οπτικοποίηση κατατμημένων οντολογιών γραμμένες σε γλώσσα OWL. Πιο συγκεκριμένα, το εργαλείο αυτό δημιουργεί γράφους αναπαράστασης των κλάσεων και των σχέσεων των οντολογιών γραμμένων σε γλώσσα OWL-DL και με εκφραστικότητα το πολύ SHIQ, ενώ συνδέει τις οντολογίες αυτές απεικονίζοντας τις αντιστοιχίσεις και τις συνδέσεις μεταξύ των στοιχείων αυτών των οντολογιών που δηλώνονται στην C-OWL. Σκοπός του Εργαλείου Οπτικοποίησης Κατατμημένων Οντολογιών είναι να δημιουργήσει ένα διαδραστικό γράφο, όπου απεικονίζονται τα αρθρώματα μιας κατακτημένης οντολογίας και τις μεταξύ τους συνδέσεις κατά το πρότυπο του τρόπου αναπαράστασης της E-SHIQ [4], ώστε να βοηθήσει στην ανάλυση και μελέτη των οντολογιών αυτών.

Η απεικόνιση μιας οντολογίας αποτελεί σημαντικό τμήμα της μελέτης/ανάλυσής της, καθώς η οπτικοποίηση της δομής και των σχέσεών της, την καθιστά πιο καταληπτή,

ιδιαίτερα δε αν είναι μεγάλη (περιέχει πολλά στοιχεία) και/ή οι συνδέσεις μεταξύ των στοιχείων της είναι ιδιαίτερα περίπλοκες. Αυτό δε ισχύει ακόμα περισσότερο στην περίπτωση των κατατμημένων οντολογιών, όπου τα αξιώματα ενός τμήματος (αρθρώματος) της οντολογίας μπορούν να επηρεάζουν την εγκυρότητα αξιωμάτων σε άλλο τμήμα της. Η οπτικοποίηση των συνδέσεων μεταξύ των τμημάτων μιας κατατμημένης οντολογίας και η περιήγηση μεταξύ αυτών είναι βασικό εργαλείο για τη μελέτη και την κατανόησή τους και επομένως στη σωστή συντήρηση, χρήση, επέκτασή τους κλπ.

Για παράδειγμα, η αποικόνιση μιας οντολογίας χρησιμοποιώντας ένα γράφο που αποτελείται από κόμβους που απεικονίζουν τις κλάσεις της οντολογίας και ακμές που απεικονίζουν τις σχέσεις μεταξύ αυτών των κλάσεων, είναι πολύ πιο ευδιάκριτες οι κλάσεις καθώς και οι σχέσεις τους συγκριτικά με μια οντολογία που είναι γραμμένη χρησιμοποιώντας XML/RDF. Στο σημείο αυτό πρέπει να προσθέσουμε ότι η πλήρης ανάλυση μιας οντολογίας απαιτεί να εξεταστεί αυτή και από το μέρος της σημασιολογίας της με τη βοήθεια των μηχανισμών αυτόματης εξαγωγής συμπερασμάτων (π.χ. Pellet, Hermit FaCT). Η χρήση όμως τέτοιων μηχανισμών για κατατμημένες οντολογίες (μόλις πρόσφατα έχουν αναπτυχθεί κάποιοι, όπως E-SHIQ και DRAGO) είναι πέρα από την εμβέλεια της παρούσης διπλωματικής. Σκοπός όμως αυτής της διπλωματικής είναι η ανάλυση της δομής και των σχέσεων που περιέχει μια οντολογία, μέσω της οπτικοποίησής της και της διάδρασης με αυτήν.

Στα παρακάτω κεφάλαια αναλύονται οι γλώσσες αναπαράστασης γνώσης στις οποίες βασίστηκε η δημιουργία της OWL, η αναπαράσταση πολλαπλών οντολογιών και η σύνδεσή τους σε μια κατατμημένη οντολογία, και κατ' επέκταση η C-OWL. Έπειτα από την ανάλυση των γλωσσών αυτών, λαμβάνει χώρα η περιγραφή της υλοποίησης του Εργαλείου Οπτικοποίησης Κατατμημένων Οντολογιών, καθώς και σενάρια χρήσης του. Εν κατακλείδι, παρατίθενται πληροφορίες για τα επικρατέστερα εργαλεία που υπάρχουν μέχρι στιγμής για την απεικόνιση οντολογιών, τα συμπεράσματα της ανάπτυξης του εργαλείου, καθώς και μελλοντικές εξελίξεις του.

2. Προαπαιτούμενες Γνώσεις

Σε αυτό το κεφάλαιο θα αναλυθούν οι τρείς κύριες γλώσσες, πάνω στις οποίες βασίστηκε η ανάλυση και η δημιουργία του εργαλείου οπτικοποίησης κατατμημένων οντολογιών. Οι γλώσσες αυτές είναι η (οικογένεια) DL (Description Logic) (Λογικές Περιγραφής) που αποτελεί το βασικό πλαίσιο αναπαράστασης γνώσης στο οποίο βασίστηκε η έκδοση της OWL που μας ενδιαφέρει στα πλαίσια της διπλωματικής αυτής, η OWL (Web Ontology Language) και η C-OWL (Contextualizing Web Ontology Language). Είναι απαραίτητη αυτή η παρουσίαση ώστε να γίνει κατανοητό το πώς το εργαλείο επεξεργάζεται και αναλύει βάση των συντακτικών κανόνων των γλωσσών αυτών τα στοιχεία των οντολογιών.

2.1. Web Ontology Language (OWL)

Σε αυτό το σημείο θα γίνει η παρουσίαση της γλώσσας OWL. Αρχικά θα γίνει μια αναφορά στο μοντέλο δεδομένων που βασίζεται η OWL και όλος ο σημασιολογικός ιστός, το RDF, και στη συνέχεια θα παρουσιαστεί το πώς εξελίχθηκε η γλώσσα στη μορφή που είναι σήμερα. Τέλος θα μελετηθούν οι κύριοι συντακτικοί και κανόνες και η σημασιολογία τους, καθώς και το πως συσχετίζονται αυτοί με τις γλώσσες περιγραφικής λογικής DL.

2.1.1. RDF, RDF-schemas ως βασικά προαπαιτούμενα της OWL

Η RDF (Resource Description Framework) παρόλο που αποκαλείται συχνά «γλώσσα», στην ουσία είναι ένα μοντέλο δεδομένων. Αποτελεί το επικρατέστερο μοντέλο στο οποίο βασίστηκαν η σύνταξη αλλά και η δημιουργία της OWL. Το RDF είναι ένα καθολικό μοντέλο δεδομένων που επιτρέπει στους χρήστες να περιγράφουν πόρους, χρησιμοποιώντας τα δικά τους λεξιλόγια. Η RDF δεν κάνει υποθέσεις σχετικά με κάποιο συγκεκριμένο πεδίο εφαρμογής, ούτε και αναπαριστά σημασιολογικά κάποιο πεδίο γνώσης. Αυτό μπορεί να γίνει με τη χρήση της γλώσσας RDF Schemas (RDFS) η οποία χρησιμοποιείται για τον ορισμό των λεξικών βάσει των οποίων περιγράφονται τα δεδομένα RDF. Το συντακτικό της RDF αποτελείται από ένα βασικό δομικό στοιχείο, την τριπλέτα αντικειμένου-χαρακτηριστικού-τιμής, η οποία ονομάζεται πρόταση (statement). Η RDF είναι ανεξάρτητη πεδίου, που σημαίνει ότι ο κάθε χρήστης είναι υπεύθυνος για τον ορισμό της δικής του ορολογίας. Η ιδιότητα αυτή δίνει ελευθερία στο χρήστη να ορίσει τις δικές του ιδιότητες και τιμές που μπορούν να πάρουν αυτές [1]. Παραδείγματος χάρη ένας χρήστης μπορεί να ορίσει ένα πόρο που θέλει να περιγράψει, μια ιδιότητα που περιγράφει το πόρο και μια τιμή για αυτή την ιδιότητα. Με αυτό τον τρόπο δημιουργεί μια μοναδική περιγραφή μιας πρότασης(π.χ. «Ο Γρηγόρης Αντωνίου είναι συγγραφέας του βιβλίου Εισαγωγή στο Σημασιολογικό Ιστό» δημιουργεί την τριπλέτα (Εισαγωγή στο Σημασιολογικό Ιστό, έχει συγγραφεί, Γρηγόρης Αντωνίου)).

Οι θεμελιώδεις έννοιες της RDF είναι οι εξής :

- **Πόρος:** Ένα οποιοδήποτε αντικείμενο, ένα πράγμα το οποίο καλείται να περιγραφεί.
- **Ιδιότητες:** Οι ιδιότητες αποτελούν μια ειδική περίπτωση πόρων που περιγράφουν σχέσεις μεταξύ πόρων
- **Προτάσεις:** Οι προτάσεις χρησιμοποιούνται για να δηλώσουν τις ιδιότητες των πόρων. Η πρόταση είναι μια τριπλέτα (triple) πόρου-χαρακτηριστικού-τιμής, η

οποία αποτελείται από έναν **πόρο** , μια **ιδιότητα** και μια **τιμή** που μπορεί να είναι πόρος ή ένα λεκτικό (literal).

2.1.2. Από την RDF στην OWL

Η δυνατότητα περιγραφής περίπλοκων αντικειμένων, καθώς και η εκφραστικότητα των γλωσσών RDF και RDF Schemas είναι περιορισμένη. Ο λόγος είναι ότι η RDF περιορίζεται σε δυαδικά κατηγορήματα (τριάδες) και σε μια μόνο ιεραρχία υποκλάσεων και μια ιεραρχία ιδιοτήτων. Τα παραπάνω, σε συνδυασμό με τον προσδιορισμό περιπτώσεων από τον οργανισμό W3C, όπου για την περιγραφή κάποιων σχέσεων μεταξύ εννοιών χρειαζόταν πολύ μεγαλύτερη εκφραστικότητα από αυτή που προσφέρουν οι RDF και RDF Schemas, γέννησαν την ανάγκη για τη δημιουργία μιας γλώσσας σημασιολογίας με μεγαλύτερη εκφραστική δύναμη. Πιο συγκεκριμένα, η RDFs επιτρέπει την αναπαράσταση ενός μέρους της οντολογικής γνώσης.

Οι κύριες δυνατότητες που λείπουν από την RDF είναι οι εξής [1]:

- Τοπική εμβέλεια τιμών (αδυναμία ορισμού περιορισμών στο σύνολο των τιμών για συγκεκριμένες κλάσεις).
- Μη επικάλυψη κλάσεων (αδυναμία ορισμού ξένων κλάσεων μεταξύ τους).
- Έλλειψη λογικών συνδυασμών κλάσεων (έλλειψη δυνατότητας ορισμού νέων κλάσεων συνδυάζοντας άλλες κλάσεις π.χ. με τελεστές «ένωσης» ή «τομής»).
- Περιορισμοί πληθικότητας ιδιοτήτων (αδυναμία ορισμού περιορισμών στο πλήθος των διακριτών τιμών που μπορεί να πάρει μια ιδιότητα).
- Έλλειψη δυνατότητας περιγραφής ειδικών χαρακτηριστικών ιδιοτήτων (π.χ. μεταβατικών ιδιοτήτων όπως ότι μια κλάση είναι "μεγαλύτερη" από μια άλλη.

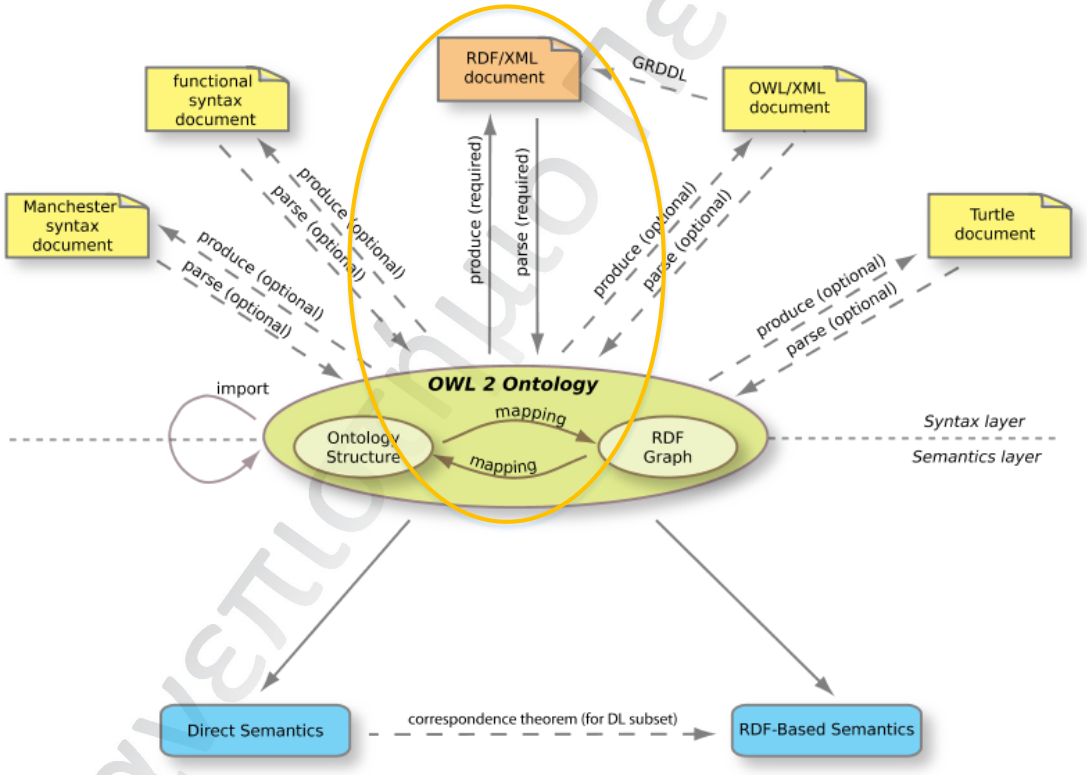
Η ομάδα Εργασιών Οντολογιών(Web Ontology Working Group W3C) έχοντας χρησιμοποιήσει την DAML+OIL ως αφετηρία και μετέπειτα βασιζόμενη στο συντακτικό της RDF, δημιούργησε την Web Ontology Language (OWL), ώστε να καλύψει τις απαιτήσεις εκφραστικότητας μιας οντολογίας.

Η OWL έχει οριστεί αρχικά ως τρεις διαφορετικές υπογλώσσες, κάθε μια σχεδιασμένη για να καλύπτει διαφορετικές πτυχές του πλήρους συνόλου των απαιτήσεων. Η πρώτη γλώσσα είναι η **OWL full** που χρησιμοποιεί όλα τα γλωσσικά θεμελιώδη στοιχεία της OWL. Η δεύτερη γλώσσα είναι η **OWL DL (**Description Logic**).** Η OWL DL είναι υπογλώσσα της OWL Full και δημιουργήθηκε για να ανακτήσει την υπολογιστική αποδοτικότητα που λείπει από την OWL full, με το να περιορίζει τον τρόπο χρήσης των «δομικών στοιχείων» (constructors) της OWL. Ουσιαστικά δεν επιτρέπεται η εφαρμογή ενός «δομικού στοιχείου» της OWL σε ένα άλλο, διασφαλίζοντας έτσι την αντιστοίχιση της γλώσσας σε μία καλά ορισμένη περιγραφική λογική. Το πλεονέκτημά της είναι ότι επιτρέπει την αποδοτική υποστήριξη συλλογισμών. Η τρίτη γλώσσα είναι η **OWL Lite** όπου είναι υπογλώσσα της OWL DL. Οι επιπρόσθετοι περιορισμοί που εισάγει η OWL Lite είναι η «απαγόρευση» των απαριθμητικών κλάσεων (enumerated classes), των προτάσεων επικάλυψης (disjointness statements) και της αυθαίρετης πληθικότητας ιδιοτήτων (arbitrary cardinality). Οι περιορισμοί αυτοί μπορεί να μειώνουν την εκφραστικότητα, αλλά κάνουν τη γλώσσα ευκολότερα κατανοητή (στους χρήστες), ευκολότερη στη χρήση για την υλοποίηση οντολογιών (για τους κατασκευαστές εργαλείων) και αποδοτικότερη στην επεξεργασία της από μηχανές αυτόματης εξαγωγής συμπερασμών[1]. Για τους

σκοπούς της παρούσας διπλωματικής εργασίας και του εργαλείου που κλήθηκε να δημιουργήσει, χρησιμοποιήθηκε η υπογλώσσα OWL-DL, καθώς αποτελεί μια γλώσσα με μεγαλύτερη εκφραστικότητα από την OWL Lite που διατηρεί παρόλα αυτά την υπολογιστική αποδοτικότητα, σε αντίθεση με την OWL full.

2.1.3. Σύνταξη γλώσσας OWL

Όπως φαίνεται και στην Εικόνα 1 υπάρχουν πολλοί τρόποι σύνταξης για την OWL όπως Manchester, RDF/XML, OWL/XML. Η υπογλώσσα OWL που χρησιμοποιείται για την δημιουργία του Εργαλείου οπτικοποίησης Κατατμημένων Οντολογιών είναι η OWL DL, και η απαίτηση είναι οι επιμέρους οντολογίες (τμήματα/αρθρώματα της κατατμημένης οντολογίας) να έχουν εκφραστικότητα το πολύ SHIQ (αυτό θα εξηγηθεί παρακάτω). Η σύνταξη που χρησιμοποιήθηκε βασίζεται στην γλώσσα RDF Schemas, χρησιμοποιώντας επίσης και τη σύνταξη που χρησιμοποιούν αυτές οι γλώσσες, που βασίζεται στην XML(Εικόνα 1). Στη συνέχεια ακολουθούν οι βασικοί συντακτικοί κανόνες για τη δημιουργία μίας OWL οντολογίας[1]: Για την ανάγνωση και την κατανόηση των στοιχείων μιας οντολογίας στα πλαίσια της παρούσης διπλωματικής αναπτύχθηκαν τα κατάλληλα εργαλεία.



Εικόνα 1: Σχήμα συνόλου συντάξεων της γλώσσας OWL 2[11].

2.1.3.1. Ορισμός Κεφαλίδας

Κάθε έγγραφο OWL ξεκινάει με μια κεφαλίδα που ορίζει το στοιχείο «ρίζα» μιας οντολογίας, που είναι ένα στοιχείο rdf:RDF το οποίο καθορίζει επίσης και έναν χώρο ονομάτων.

```
<rdf:RDF
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#">
```

2.1.3.2. Ορισμός κλάσεων και αξιωμάτων

Οι κλάσεις αποτελούν ένα από τα δομικά στοιχεία μιας οντολογίας˙ για να ορίσουμε μια κλάση χρησιμοποιούμε το στοιχείο **owl:class**.

```
<owl:Class rdf:ID="#man">
```

Μια από τις βασικότερες σχέσεις μέσα σε μια οντολογία είναι οι σχέσεις ιεραρχίας των κλάσεων(σχέσεις υποκλάσεων). Για να ορίσουμε μια κλάση είναι υποκλάση μιας άλλης χρησιμοποιούμε το στοιχείο **owl:subClassOf.** Στο παρακάτω κομμάτι κώδικα παρουσιάζεται το πώς ορίζεται ότι η κλάση man είναι υποκλάση της human.

```
<owl:Class rdf:ID="#man">
<owl:subClassOf rdf:resource="#human"/>
</owl:Class>
```

Η OWL επίσης δίνει τη δυνατότητα καθορισμού σχέσεων «ξενικότητας» μεταξύ κλάσεων χρησιμοποιώντας το στοιχείο **owl:disjointwith**. Παραδείγματος χάρι η κλάση woman είναι ξένη με τη κλάση man.

```
<owl:Class rdf:about="#human">
<owl:disjointWith rdf:resource="#woman"/>
<owl:disjointWith rdf:resource="#man"/>
</owl:Class>
```

Αντίστοιχα για να ορίσουμε ισοδύναμες κλάσεις χρησιμοποιούμε το στοιχείο **owl:equivalentClass**.

```
<owl:Class rdf:ID="person">
<owl:equivalentClass rdf:resource="#human"/>
</owl:Class>
```

2.1.3.3. Ορισμός ιδιοτήτων

Υπάρχουν δυο είδη ιδιοτήτων στην OWL. Το πρώτο είδος ιδιοτήτων αφορά τις σχέσεις μεταξύ αντικειμένων (στιγμιοτύπων κλάσεων οντολογίας) και ονομάζονται ιδιότητες αντικειμένων (object property). Οι ιδιότητες αντικειμένων συσχετίζουν κλάσεις με άλλες κλάσεις χρησιμοποιώντας το στοιχείο **owl:ObjectProperty**.

```
<owl:ObjectProperty rdf:ID="married ">
<rdfs:domain rdf:resource="#man"/>
<rdfs:range rdf:resource="#woman"/>
<rdfs:subPropertyOf rdf:resource="#involves"/>
</owl:ObjectProperty>
```

Όπως φαίνεται και στο κώδικα μια ιδιότητα αντικειμένων έχει χαρακτηριστικά που την ορίζουν. Τα χαρακτηριστικά αυτά είναι τα εξής:

- **rdfs:domain** (πια κλάση ορίζει την ιδιότητα, ή ποιά κλάση χαρακτηρίζεται από την ιδιότητα)
- **rdfs:range** (πια κλάση δίνει τιμές στην ιδιότητα)
- **rdfs:super** (πια είναι η υπερ-ιδιότητα της ιδιότητα)

Η OWL επίσης επιτρέπει τις σχέσεις μεταξύ των ιδιοτήτων. Οι σχέσεις μεταξύ των ιδιοτήτων είναι οι εξής:

- **rdfs:subpropertyOf** (υπο-ιδιότητα κάποιας άλλης ιδιότητας)
- **rdfs:inverseOf** (αντίστροφη ιδιότητα κάποιας άλλης ιδιότητας)

Επίσης η OWL παρέχει την δυνατότητα να οριστούν και ιδιότητες για τις ίδιες τις ιδιότητες, οι βασικές ιδιότητες ιδιοτήτων είναι:

- **rdfs:symmetric** (συμμετρική ιδιότητα κάποιας άλλης ιδιότητας)
- **rdfs:transitive** (μεταβατική ιδιότητα κάποιας άλλης ιδιότητας)

Το δεύτερο είδος ιδιοτήτων είναι οι ιδιότητες τύπου δεδομένων **(owl:DatatypeProperty)**, οι οποίες συσχετίζουν αντικείμενα με τιμές ενός τύπου δεδομένων. Για παράδειγμα η ιδιότητα age που φαίνεται στο παρακάτω κομμάτι κώδικα έχει το περιορισμό ότι μπορεί να πάρει τιμές μη αρνητικούς ακέραιους αριθμούς.

```
<owl:DatatypeProperty rdf:ID="age">
<rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema
#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

2.1.3.4. Ορισμός περιορισμών ιδιοτήτων

Οι περιορισμοί ιδιοτήτων χρησιμοποιούνται για τον ορισμό κλάσεων αντικειμένων και εκφράζουν περιορισμούς για τις τιμές ιδιοτήτων για τα αντικείμενα της κλάσης αυτής χρησιμοποιώντας το στοιχείο **owl:restriction**. Στο κομμάτι κώδικα που ακολουθεί ορίζεται πως η κλάση married_man παίρνει όλες τις τιμές της από τη κλάση man.

```
<owl:Class rdf:about="#married_man">
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#married"/>
<owl:allValuesFrom rdf:resource="#man"/>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

Οι τύποι των περιορισμών είναι οι εξής:

- **Owl:someValuesFrom**(μια ιδιότητα παίρνει τουλάχιστον μια τιμή από μια κλάση)
- **Owl:allValuesFrom**(μια ιδιότητα παίρνει τιμές μόνο από μια κλάση)
- **Owl:hasValue**(μια ιδιότητα έχει συγκεκριμένη τιμή από μια κλάση)
- **Owl:minCardinality**(μια ιδιότητα έχει αριθμητικό περιορισμό ελάχιστου αριθμού τιμών)
- **Owl:maxCardinality**(μια κλάση έχει αριθμητικό περιορισμό μεγίστου αριθμού τιμών)

2.1.3.5. Ορισμός λογικών συνδυασμών

Η OWL δίνει τη δυνατότητα ορισμού κλάσεων που αποτελούν λογικούς συνδυασμούς άλλων κλάσεων, όπως η ένωση, η τομή και το συμπλήρωμα κλάσεων. Όπως φαίνεται και στο παράδειγμα η κλάση human αποτελεί ένωση των κλάσεων man και woman.

```
<owl:Class rdf:ID="human">
<owl:unionOf rdf:parseType="Collection">
```

```
<owl:Class rdf:about="#man"/>
<owl:Class rdf:about="#woman"/>
</owl:unionOf>
</owl:Class>
```

Οι λογικοί συνδυασμοί είναι οι εξής:

- Owl:unionOf(ένωση κλάσεων)
- Owl:intersectionOf (τομή κλάσεων)
- Owl:complementOf (συμπλήρωμα κλάσης)
- Owl:one of (τιμή από μια συλλογή κλάσεων)

2.2. Description Logic (DL)

Η γλώσσα DL αποτελεί τον επίσημο φορμαλιστικό και μαθηματικό υπόβαθρο για την ανάπτυξη γλωσσών οντολογιών. Η OWL DL βασίστηκε στη DL για τη ανάπτυξη των κανόνων που την διέπουν. Όντας η OWL η κυρίαρχη γλώσσα του σημασιολογικού ιστού, είναι σημαντικό να αναλυθούν οι βάσεις στις οποίες στηρίχτηκε. Με αυτό τον τρόπο θα γίνει καλύτερα κατανοητός ο τρόπος με τον οποίο περιγράφονται τα δεδομένα, καθώς και η αναφορά στην εκφραστικότητα των γλωσσών αναπαράστασης, αλλά και η αναφορά στη σύνδεση οντολογιών. Στα επόμενα κεφάλαια ακολουθεί η παρουσίαση των δομικών στοιχείων των λογικών περιγραφής (ή περιγραφικών λογικών) και της συσχέτισής τους με τα δομικά στοιχεία της OWL .

2.2.1. Περιγραφικές Λογικές

Οι περιγραφικές λογικές (Description Logic (DL)) αποτελούν μια οικογένεια επίσημων γλωσσών για την αναπαράσταση της γνώσης με βάση τις έννοιες ενός πεδίου γνώσης (εννοιολογική αναπαράσταση). Στις γλώσσες DL για να μπορέσει να δημιουργηθεί η περιγραφή ενός αντικειμένου πρώτα ορίζονται τα «βασικά» στοιχεία που το περιγράφουν (ορολογία (terminology)) μέσω της ονομασίας κλάσεων και ιδιοτήτων και στη συνέχεια χρησιμοποιούνται αυτά ώστε να καθοριστούν πολυπλοκότεροι ορισμοί στοιχείων που προκύπτουν είτε μέσω καθορισμού σχέσεων μεταξύ των στοιχείων (μέσω ιδιοτήτων) είτε μέσω συνδυασμού στοιχείων (π.χ. ένωσης, τομής κλπ. κλάσεων)[5].

Ένα από τα κύρια χαρακτηριστικά αυτής της οικογένειας γλωσσών, σε αντίθεση με πολλούς προκατόχους της, είναι η επίσημη, βασισμένη σε λογικούς κανόνες σημασιολογία. Αυτό δίνει τη δυνατότητα χρήσης της γλώσσας από μηχανές αυτόματης εξαγωγής συμπερασμάτων (reasoners) για την διενέργεια συλλογιστικής (π.χ για την ταξινόμηση νέων ορισμών κλάσεων αντικειμένων ή για την κατηγοριοποίηση αντικειμένων σε κλάσεις)

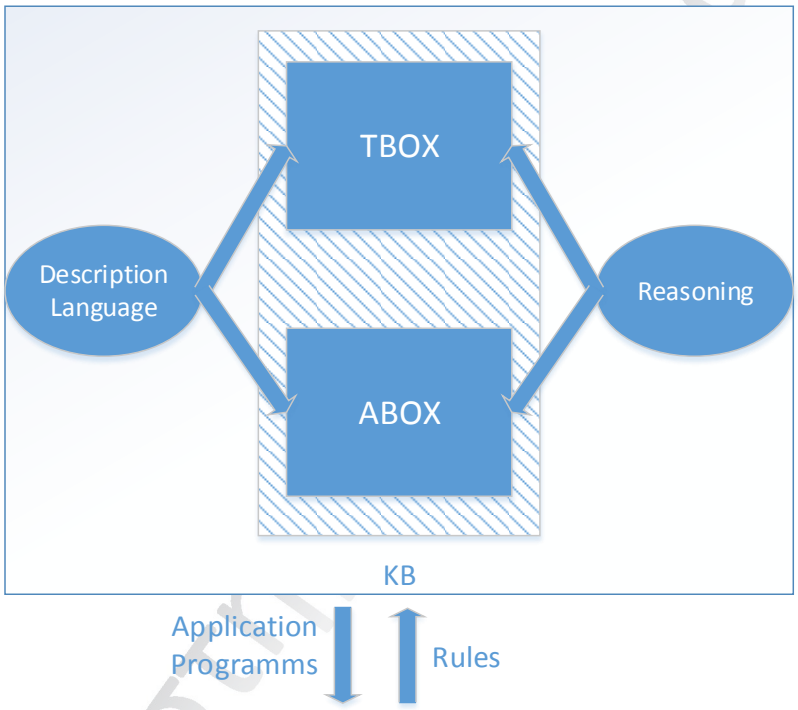
Η οικογένεια γλωσσών DL προέρχεται από τα “structured inheritance networks”, που δημιουργήθηκαν για να εξαλείψουν την ασάφεια στα πρώτα σημασιολογικά δίκτυα και αποτελούν υποσύνολο της λογικής πρώτης τάξης. Τρία χαρακτηριστικά κληρονομήθηκαν στην DL ώστε να διασφαλίσουν αυτή την ιδιότητα:

- Τα βασικά συντακτικά στοιχεία είναι είτε ατομικές έννοιες, είτε ατομικές ιδιότητες, είτε στιγμιότυπα εννοιών.

- Η εκφραστική δύναμη της γλώσσας περιορίζεται στο να χρησιμοποιεί ένα αριθμό δομικών στοιχείων για να ορίζει πολύπλοκα αντικείμενα και ιδιότητες αντικειμένων μέσω αυτών.
- Η προφανής πληροφορία που αφορά τις έννοιες ή τα στιγμιότυπα μπορεί να συναχθεί αυτόματα με τη βοήθεια διαδικασιών παραγωγής συμπερασμάτων.

2.2.2. Σύνταξη γλώσσας DL και χαρακτηριστικά

Μια βάση γνώσης DL (και κατά βάση οποιαδήποτε οντολογία) περιέχει δύο διακριτά τμήματα εκ των οποίων το πρώτο αφορά τον ορισμό των εννοιών (κλάσεων) και των σχέσεών τους (Terminology Box(TBox)), ενώ το δεύτερο, αφορά τα στιγμιότυπα των κλάσεων και των ιδιοτήτων (Assertion Box (ABox))(Εικόνα 2)[5] .



Εικόνα 2: Αρχιτεκτονική συστήματος αναπαράστασης γνώσης (Knowledge Representation) βασισμένη στη DL.

2.2.2.1. Εννοιολογικό κουτί (TBox)

Οι βασικοί συντακτικοί κανόνες που μπορούν να χρησιμοποιηθούν για τον καθορισμό κλάσεων (εννοιών) χρησιμοποιώντας την βασική γλώσσα *AL* της οικογένειας των περιγραφικών λογικών είναι οι εξής

C, D → A| (ατομική-βασική- έννοια (atomic Concept))
⊤| (καθολική έννοια (universal concept))
⊥| (εδραία έννοια (bottom concept))
¬A| (ατομική άρνηση (atomic negation)
C⊓D| (τομή (intersection))
∀R.C| (περιορισμός τιμής (Value restriction))
∃ R. ⊤ | (περιορισμένος υπαρξιακός προσδιορισμός (limited existential quantification))

Για να γίνουν προτάσεις οι οποίες αφορούν το πώς «πολυπλοκότερες» έννοιες ορίζονται και συσχετίζονται με άλλες έννοιες ή σχέσεις, οι περιγραφικές λογικές εισάγουν τα λεγόμενα αξιώματα ορολογίας (terminological axioms). Στην πιο γενική μορφή τους τα αξιώματα αυτά έχουν αυτή τη μορφή: C ⊑D (R ⊑S) ή C ≡ D (R ≡ S). Όπου C, D είναι έννοιες και R,S είναι σχέσεις. Τα σύμβολά ⊑ υποδηλώνουν ότι η μια έννοια είναι υποσύνολο της άλλης (C υποσύνολο (υποκλάση του) D) και τα σύμβολα ≡ υποδηλώνουν ότι οι δύο έννοιες ταυτίζονται (C ταυτίζεται με D). Τα σύμβολα αυτά έχουν την ίδια ερμηνεία και για τις σχέσεις

2.2.2.2. Κουτί Δηλώσεων (ABox)

Όπως αναφέραμε και στην αρχή του κεφαλαίου 2.2.2, το ABox περιέχει στιγμιότυπα των κλάσεων και των σχέσεων που έχουν οριστεί στο TBox. Για να εισαχθεί ένα στιγμιότυπο στο ABox πρέπει να κατηγοριοποιηθεί κατάλληλα υπό μια κλάση ως ένα συγκεκριμένο στιγμιότυπο αυτής. Για να γίνει πιο κατανοητό, ακολουθεί ένα παράδειγμα κάποιων στιγμιότυπων που αφορούν το TBox [5].

- MotherWithoutDaughter(MARY) Father(PETER)
- hasChild(MARY; PETER) hasChild(PETER; HARRY)
- hasChild(MARY; PAUL)

2.2.3. Η γλώσσα περιγραφής SHIQ

Όπως έχει προαναφερθεί στην αρχή του κεφαλαίου, η DL αποτελεί μια οικογένεια γλωσσών, εκ των οποίων κάθε μια έχει δημιουργηθεί για να καλύπτει διαφορετικές εκφραστικές ανάγκες. Η DL γλώσσα πάνω στην οποία βασίστηκε η δημιουργία της γλώσσας OWL-DL, είναι η SHOIN(D), η οποία υποστηρίζει τις εξής επιπλέον πράξεις από τη βασική DL[5]:

- Υπαρξιακούς περιορισμούς(Limited existential quantification)
- Άρνηση περίπλοκων εννοιών(Complex concept negation)
- Ιεραρχία ρόλων(Role hierarchy)
- Ορισμό ανάστροφων ρόλων(Inverse properties)
- Ποσοτικούς περιορισμούς(Qualified cardinality restrictions)
- Μεταβατικούς ρόλους(Transitive roles)

Στα πλαίσια της διπλωματικής εργασίας αυτής περιοριζόμαστε στη χρησμιμοποίηση της SHIQ όπου αποτελεί υπογλώσσα της SHOIN(D). Η SHIQ έχει τους παρακάτω συντακτικούς κανόνες που ενισχύουν την εκφραστική της δύναμη σε σχέση με την *AL:*

$\{x_1,\ldots x_n\}$| τιμή από μια συλλογή κλάσεων(one of)
$\exists R.\{x\}$| ιδιότητα με συγκεκριμένη τιμή από μια κλάση (has value)
R^- | αντίστροφη ιδιότητα κάποιας άλλης ιδιότητας (inverse of)

2.2.4. Σχέση OWL με DL

Για να γίνει πλήρως αντιληπτή η συσχέτιση της OWL με τη DL, παρατίθεται ο πίνακας της Εικόνας 3 που απεικονίζει την πλήρη συντακτική αντιστοιχία[5].

OWL Abstract Syntax	DL syntax
Class axioms	
Class(A partial C_1 ... C_n)	$A \sqsubseteq C_i$
Class(A complete C_1 ... C_n)	$A \equiv C_1 \sqcap \ldots \sqcap C_n$
EnumeratedClass(A o_1 ... o_n)	$A \equiv \{o_1, \ldots o_n\}$
SubClassOf(C_1 C_2)	$C_1 \sqsubseteq C_2$
EquivalentClasses(C_1 ... C_n)	$C_1 \equiv \ldots \equiv C_n$
DisjointClasses(C_1 ... C_n)	$C_i \sqcap C_j \sqsubseteq \bot$
Property axioms	
ObjectProperty(R	
super(R_1)...super(R_n)	$R \sqsubseteq R_i$
domain(C_1) ... domain(C_n)	$\top \sqsubseteq \forall R^-.C_i$
range(C_1) ... range(C_n)	$\top \sqsubseteq \forall R.C_i$
[inverseOf(R_0)]	$R \equiv R_0^-$
[Symmetric]	$R \equiv R^-$
[Functional]	$\top \sqsubseteq \leqslant 1R$
[InverseFunctional]	$\top \sqsubseteq \leqslant 1R^-$
[Transitive])	Trans(R)
Datatype(T)	
DatatypeProperty(U	
super(U_1)...super(U_n)	$U \sqsubseteq U_i$
domain(C_1) ... domain(C_n)	$\top \sqsubseteq \forall U^-.C_i$
range(T_1) ... range(T_n)	$\top \sqsubseteq \forall U.T_i$
[Functional])	$\top \sqsubseteq \leqslant 1U$
SubPropertyOf(Q_1 Q_2)	$Q_1 \sqsubseteq Q_2$
EquivalentProperties(Q_1 ... Q_n)	$Q_1 \equiv \ldots \equiv Q_n$

OWL Abstract Syntax	DL syntax
A (URI Reference)	A
owl:Thing	$\top$
owl:Nothing	$\bot$
intersectionOf(C_1 ... C_n)	$C_1 \sqcap \ldots \sqcap C_n$
unionOf(C_1 ... C_n)	$C_1 \sqcup \ldots \sqcup C_n$
complementOf(C)	$\neg C$
oneOf(o_1 ... o_n)	$\{o_1, \ldots o_n\}$
restriction(R someValuesFrom(C))	$\exists R.D$
restriction(R allValuesFrom(C))	$\forall R.D$
restriction(R value(o))	$\exists R.o$
restriction(R minCardinality(n))	$\geqslant nR$
restriction(R maxCardinality(n))	$\leqslant nR$
restriction(U someValuesFrom(T))	$\exists U.T$
restriction(U allValuesFrom(T))	$\forall U.T$
restriction(U value(t))	$\exists U.t$
restriction(U minCardinality(n))	$\geqslant nU$
restriction(U maxCardinality(n))	$\leqslant nU$

Εικόνα 3: Πίνακας μετάφρασης του συντακτικού OWL-DL σε συντακτικό DL[5].

Όπως φαίνεται και στην Εικόνα 3 και λαμβάνοντας υπόψιν την σύνταξη που έχει αναλυθεί στο κεφάλαιο της OWL και της DL μπορεί να γίνει εύκολα αντιληπτή η σύνδεση της OWL και της DL. Αρχικά παρατίθενται τα αξιώματα που αφορούν τις σχέσεις μεταξύ κλάσεων.

- $C_1 \sqsubseteq C_2$: Σχέση υποκλάσης (`owl:subClassOf`)
- $C_1 \equiv C_2$: Σχέση ισοδύναμων κλάσεων(`owl:equivalentClass`)
- $C_1 \sqcap C_2 \sqsubseteq \bot$: Σχέση ξένων κλάσεων(`owl:disjointWith`)
- $A \sqsubseteq C_1$: Απλός ορισμός κλάσης(`owl:class`)

Στην συνέχεια παρατίθενται τα αξιώματα που αφορούν τις ιδιότητες και τις σχέσεις τους.

- $R \sqsubseteq S$: Ορισμός υπερ-ιδιότητας (owl:subpropertieOf)
- $\top \sqsubseteq \forall R^-.C$: Ορισμός πεδίου ορισμού (owl:domain)
- $\top \sqsubseteq \forall R.C$:Ορισμός πεδίου τιμών (owl:range)
- $R \equiv S^-$: Ανάστροφη ιδιότητα (owl:inverseOf)
- $R \equiv R^-$: Συμμετρική ιδιότητα (owl:symmetric)
- Trans(R):Μεταβατική ιδιότητα (owl:transitive)

Ακολουθούν οι σχέσεις που αφορούν συνδυασμούς κλάσεων (ένωση, τομή, κλπ.)

- $C_1 \sqcap \ldots \sqcap C_n$: Τομή κλάσεων (`owl:intersectionOf`)
- $C_1 \sqcup \ldots \sqcup C_n$: Ένωση κλάσεων (`owl:unionOf`)
- $\neg C$: Συμπλήρωμα κλάσεων (`owl:completmentOf`)
- $\{o_1, \ldots, o_n\}$: Τιμή από κλάση (`owl:oneOf`)

Στο τέλος παρατίθενται το πως μεταφράζονται οι περιορισμοί.

- ∃ R.D : Η ιδιότητα R παίρνει τουλάχιστον μια τιμή από μια κλάση (`owl:someValuesFrom`)
- ∀R.D: Η ιδιότητα R παίρνει τιμές μόνο από μια κλάση (`owl: allValuesFrom`)
- ∃R.o : Η ιδιότητα R παίρνει συγκεκριμένη τιμή ο (`owl: hasValuesFrom`)
- ≥nR : Ο ελάχιστος αριθμός τιμών της R μπορεί να είναι (`owl:minCardinality`)
- ≤nR: Ο μέγιστος αριθμός τιμών της R μπορεί να είναι (`owl: maxCardinality`)

Ακολουθεί ένα παράδειγμα του πώς εκφράσεις της DL μπορούν να εκφραστούν με OWL-DL συντακτικό ώστε να γίνει πιο κατανοητή η αντιστοιχία και η σύνδεση αυτών των δύο γλωσσών.

- Woman ≡ Person ⊓Female (το στοιχείο γυναίκα ταυτίζεται με το σύνολο των ανθρώπων που είναι θηλυκού γένους):

```
<owl:Class rdf:ID="woman">
        <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#person"/>
                <owl:Class rdf:about="#female"/>
        </owl:intersectionOf>
</owl:Class>
```

- Man ≡ Person ⊓ ¬Woman (το στοιχείο άνδρας ταυτίζεται με το σύνολο των ανθρώπων που δεν ανήκουν στο σύνολο των γυναικών)

```
<owl:Class rdf:ID="man">
        <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#person"/>
                <owl:complementOf rdf:resource= "#woman"/>
        </owl:intersectionOf>
</owl:Class>
```

- Mother ≡ Woman ⊓ ∃hasChild:Person (το στοιχείο μητέρα ταυτίζεται με το σύνολο των γυναικών που έχουν ένα παιδί τουλάχιστον)

```
<owl:Class rdf:ID="mother ">
<owl:intersectionOf rdf:parseType="Collection">
                <owl:complementOf rdf:resource= "#woman"/>
  <owl:Restriction>
                <owl:onProperty rdf:resource="#haschild"/>
                <owl: SomeValuesFrom rdf:resource="#person"/>
</owl:Restriction>
</owl:intersectionOf>
</owl:Class>
```

- Father ≡ Man ⊓ ∃hasChild:Person (το στοιχείο πατέρας ταυτίζεται με το σύνολο των ανδρών που έχουν ένα παιδί τουλάχιστον)

```
<owl:Class rdf:ID="father ">
```

```
<owl:intersectionOf rdf:parseType="Collection">
                    <owl:complementOf rdf:resource= "#man"/>
  <owl:Restriction>
                    <owl:onProperty rdf:resource="#haschild"/>
                    <owl: SomeValuesFrom rdf:resource="#person"/>
</owl:Restriction>
</owl:intersectionOf>
</owl:Class>
```

- Parent ≡ Father ⊔ Mother (το στοιχείο γονέας ταυτίζεται με την ένωση των στοιχείων πατέρας και μητέρα)

```
<owl:Class rdf:ID="parent">
        <owl:unionOf rdf:parseType="Collection">
                    <owl:Class rdf:about="#father"/>
                    <owl:Class rdf:about="#mother"/>
        </owl:intersectionOf>
</owl:Class>
```

- Grandmother ≡ Mother ⊓ ∃hasChild:Parent(το στοιχείο γιαγιά ταυτίζεται με το στοιχείο μητέρα όπου έχει παιδί ένα γονέα)

```
<owl:Class rdf:ID="grandmother ">
<owl:intersectionOf rdf:parseType="Collection">
                    <owl:Class rdf:resource= "#mother"/>
  <owl:Restriction>
                    <owl:onProperty rdf:resource="#haschild"/>
                    <owl:SomeValuesFrom rdf:resource="#parent"/>
</owl:Restriction>
</owl:intersectionOf>
</owl:Class>
```

- MotherWithManyChildren ≡ Mother ⊓ >3 hasChild (το στοιχείο πολύτεκνη μητέρα ταυτίζεται με το στοιχείο μητέρα με αριθμό παιδιών ανώτερο του τρία)

```
<owl:Class rdf:ID="grandmother ">
<owl:intersectionOf rdf:parseType="Collection">
                    <owl:Class rdf:resource= "#mother"/>
<owl:Restriction>
                        <owl:onProperty rdf:resource="#haschild"/>
                        <owl:minCardinality rdf:datatype= "&xsd;nonNegativeInteger">3</owl:minCardinality>
                    </owl:Restriction>
</owl:intersectionOf>
</owl:Class>
```

- MotherWithoutDaughter ≡ Mother ⊓ ∀ hasChild::Woman(το στοιχείο μητέρα με κόρη ταυτίζεται με το στοιχείο μητέρα όπου έχει παιδί γυναίκα)

```
<owl:Class rdf:ID=" MotherWithoutDaughter ">
<owl:intersectionOf rdf:parseType="Collection">
                    <owl:Class rdf:resource= "#mother"/>
<owl:Restriction>
<owl:onProperty rdf:resource="#haschild"/>
<owl:allValuesFrom rdf:resource="#man"/>
</owl:Restriction>
</owl:intersectionOf>
</owl:Class>
```

- Wife ≡ Woman ⊓ ∃ hasHusband:Man (το στοιχείο σύζυγος ταυτίζετε με το στοιχείο γυναίκα ο που έχει σύζυγο έναν άνδρα)

```
<owl:Class rdf:ID="wife ">
<owl:intersectionOf rdf:parseType="Collection">
                    <owl:complementOf rdf:resource= "#woman"/>
  <owl:Restriction>
                    <owl:onProperty rdf:resource="#hashusband"/>
                    <owl:SomeValuesFrom rdf:resource="#man"/>
</owl:Restriction>
</owl:intersectionOf>
</owl:Class>
```

2.3. Contextualizing-OWL(C-OWL)

Η γλώσσα C-OWL(Context OWL) είναι η τελευταία γλώσσα που θα μελετηθεί και αφορά στη δήλωση αντιστοιχιών μεταξύ εννοιών διαφορετικών οντολογιών. Στα παρακάτω κεφάλαια θα αναλυθούν ο σχεδιασμός καθώς και το συντακτικό της C-OWL, ώστε να γίνει κατανοητός ο τρόπος με τον οποίο η γλώσσα αυτή καθιστά δυνατούς τους συσχετισμούς μεταξύ οντολογιών.

2.3.1. Ανάλυση σχεδιασμού C-OWL

Η C-OWL είναι μια γλώσσα που αναπτύχθηκε για να υποστηρίξει την δημιουργία συσχετίσεων μεταξύ οντολογιών. Αποτελεί μια επέκταση του συντακτικού και της σημασιολογίας της OWL. Εν' αντιθέσει με την OWL η οποία δημιουργεί οντολογίες, η C-OWL θεωρεί την κάθε οντολογία ως ένα πλαίσιο (context) δηλώσεων για ένα γνωστικό αντικείμενο που συσχετίζεται με άλλα πλαίσια τα οποία έχουν οριστεί από διαφορετικά μοντέλα-οντολογίες.

Όσον αφορά την εκφραστική δύναμη, η κύρια διαφορά της θεώρησης των πλαισίων με τις οντολογίες είναι ότι στις οντολογίες δεν υποστηρίζονται οι **υποκειμενικές** συσχετίσεις διαφορετικών κλάσεων, ιδιοτήτων. Η δυνατότητα των συσχετίσεων μεταξύ στοιχείων οντολογίας φτάνει μέχρι τα όρια της οντολογίας, πιο συγκεκριμένα οι σχέσεις μεταξύ

εννοιών μπορούν να υπάρχουν μόνο αν αυτές οι έννοιες βρίσκονται στην ίδια οντολογία. Σε αντίθεση, στα πλαίσια υπάρχουν συσχετίσεις που αφορούν **υποκειμενικές** σχέσεις που υπάρχουν μεταξύ στοιχείων διαφορετικών οντολογιών.

Για να γίνει πιο ξεκάθαρο το γιατί η C-OWL επεκτείνει τις δυνατότητες της OWL, παρατίθενται κάποια παραδείγματα στα οποία η εκφραστική δύναμη της OWL δεν επαρκεί για να τα περιγράψει [3].

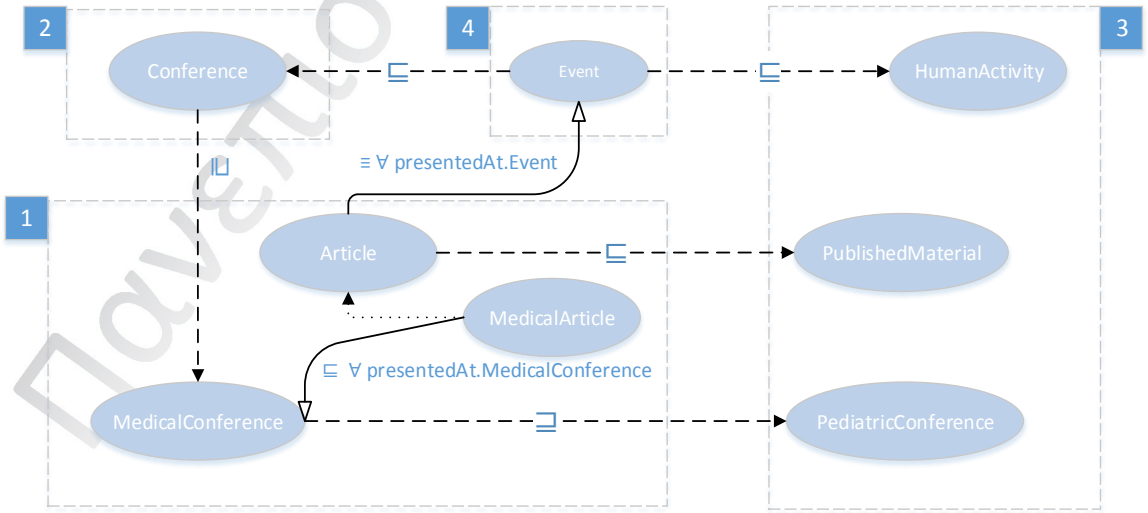
- Η κατεύθυνση των συσχετίσεων (υποκειμενικότητα): υπάρχουν περιπτώσεις που χρειάζεται να γνωρίζουμε από ποια οντολογία πηγάζει μια αντιστοιχία (source ontology) και σε ποια οντολογία καταλήγει (target ontology).
- Τοπική ονοματολογία: Χρειάζεται σε πολλές περιπτώσεις να απορριφθεί η υπόθεση ότι όλες οι οντολογίες μεταφράζονται σε μια μοναδική καθολική οντολογία.
- Πλαίσια απεικόνισης: Πρέπει να υπάρχει η δυνατότητα να οριστεί ότι δύο στοιχεία δύο διαφορετικών οντολογιών, που μπορεί να θεωρούνται διαφορετικά, περιγράφουν τον ίδιο πόρο (κλάση ή αντικείμενο) [3].

2.3.2. Σύνταξη γλώσσας C-OWL

Στη C-OWL αρχικά υπήρχε σύνδεση οντολογιών μέσω της σχέσης υποκλάσης μεταξύ κλάσεων. Πιο αναλυτικά υπήρχε η δυνατότητα να οριστεί ότι μια κλάση κάποιας οντολογίας είναι υποκλάση μιας κλάσης άλλης οντολογίας. Στην συνέχεια έγινε μια επέκταση της C-OWL από το E − SHIQ Representation Framework[4] που επιτρέπει πλουσιότερη συσχέτιση οντολογιών. Πιο συγκεκριμένα επιτρέπει τη σύνδεση μέσω, ιδιοτήτων-ρόλων και σχέσεων μεταξύ των εννοιών διαφορετικών οντολογιών.

2.3.2.1. Γράφος απεικόνισης C-OWL σχέσεων

Για να γίνει πιο ξεκάθαρος ο ρόλος της C-OWL στην σύνδεση μεταξύ οντολογιών παρατίθεται η παρακάτω Εικόνα 4, που παρουσιάζει πώς η C-OWL δημιουργεί σχέσεις υποκλάσεων(⊑) και σχέσεις ιδιοτήτων-ρόλων μεταξύ οντολογιών[4].



Εικόνα 4: Γράφος σχέσεων μεταξύ οντολογιών χρησιμοποιώντας την γλώσσα C-OWL.

Όπως φαίνεται και στην Εικόνα 4 υπάρχουν τέσσερεις διαφορετικές οντολογίες. Οι οντολογίες αυτές αποτελούν αυτοτελή μοντέλα όπου συσχετίζονται μεταξύ τους

χρησιμοποιώντας ένα πλαίσιο C-OWL. Στο σχήμα αυτό υπάρχουν δυο τύποι σχέσεων. Ο πρώτος τύπος σχέσης είναι η σχέση υποκλάσης (⊑). Η φορά του συμβόλου ⊑ δείχνει ποια κλάση θεωρείται υποκλάση κάποιας άλλης ενώ η φορά του βέλους δείχνει την υποκειμενικότητα της σύνδεσης (πια οντολογία θεωρεί ότι έχει σχέση με κάποια άλλη), π.χ. η κλάση «Event» της τέταρτης οντολογίας θεωρεί ότι είναι υποκλάση της κλάσης «HymanActivity» της τρίτης οντολογίας. Όπως προαναφέρθηκε και στη προηγούμενη παράγραφο με την βοηθείας του E − SHIQ Representation Framework γίνεται δυνατή η συσχέτιση οντολογιών και μέσω ιδιοτήτων, π.χ. η κλάση «Article» της δεύτερης οντολογίας θεωρεί ότι συνδέεται μέσο της ιδιότητας «presentedAt.Event» με την κλάση «Event» της τρίτης οντολογίας.

2.3.2.2. Σχέσεις υποκλάσεων

Στις σχέσεις υποκλάσεων δίνεται η δυνατότητα να οριστεί ότι μια κλάση μιας οντολογίας θεωρείται υποκλάση μιας άλλης οντολογίας (Εικόνα 4). Μια τέτοια απεικόνιση εκφράζεται με τον εξής τρόπο στην C-OWL. Αρχικά όλες οι σχέσεις υποκλάσεων εσωκλείονται μέσα σε ένα στοιχείο **cowl:mapping**. Στην συνέχεια ορίζονται οι οντολογίες που αφορά αυτή η απεικόνιση μέσα σε ένα στοιχείο **cowl:sourceOntology** και σε ένα στοιχείο **cowl:targetOntology** για την κλάση της μιας οντολογίας και της άλλης αντίστοιχα. Αυτά τα δύο στοιχεία (cowl:sourceOntology, cowl:targetOntology) δείχνουν επίσης και την υποκειμενικότητα της συσχέτισης, δηλαδή πιά οντολογία(cowl:sourceOntology) πιστεύει ότι έχει σχέση με κάποια άλλη (cowl:targetOntology). Έπεται ο κανόνας συσχέτισης μεταξύ των κλάσεων των οντολογιών χρησιμοποιώντας το στοιχείο **cowl:bridgeRule,** όπου μέσα εσωκλείονται η κλάση που προέρχεται από τη μία οντολογία και η κλάση που θεωρείται υποκλάση της από την άλλη οντολογία. Σημαντικό είναι να σημειωθεί ότι για τον ορισμό των κλάσεων χρησιμοποιούνται XML Schemas[3].

```
<cowl:Mapping>
  <cowl:sourceOntology>
    <owl:Ontology rdf:about="http://localhost/figure4/unit2.owl"/>
  </cowl:sourceOntology>
  <cowl:targetOntology>
    <owl:Ontology rdf:about="http://localhost/figure4/unit1.owl"/>
  </cowl:targetOntology>
  <cowl:bridgeRule>
    <cowl:Onto><cowl:source>
      <owl:Class rdf:about="http://localhost/figure4/unit2.owl#Conference"/>
    </cowl:source><cowl:target>
      <owl:Class rdf:about="http://localhost/figure4/unit1.owl#MedicalConference"/>
    </cowl:target></cowl:Onto>
  </cowl:bridgeRule>
</cowl:Mapping>
```

2.3.2.3. Σχέσεις ιδιοτήτων(E-SHIQ framework)

Στις σχέσεις ιδιοτήτων ορίζεται το πώς μια ιδιότητα μπορεί να συσχετίσει κλάσεις διαφορετικών οντολογιών (Εικόνα 4). Χρησιμοποιώντας το στοιχείο **cowl:Linking** ορίζεται ποια ιδιότητα θεωρείται από τη πρώτη οντολογία ότι αφορά μια κλάση της δεύτερης οντολογίας. Εν συνεχεία, χρησιμοποιώντας OWL Schemas οι περιορισμοί (*owl:Restriction*) μιας ιδιότητας από την μια στην άλλη οντολογία [3].

```
<cowl:Linking>
  <cowl:LinkProperty rdf:resource="http://localhost/figure4/unit1.owl#presentedAt"/>
</cowl:Linking>
```

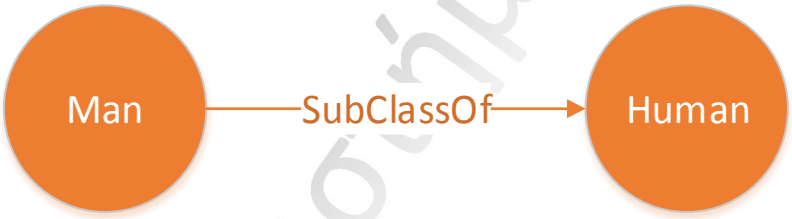
```
<owl:Class rdf:about="http://localhost/figure4/unit1.owl#Article">
 <owl:equivalentClass>
  <owl:Restriction>
   <owl:onProperty rdf:resource="http://localhost/figure4/unit1.owl#presentedAt"/>
   <owl:allValuesFrom rdf:resource="http://localhost/figure4/unit4.owl#Event"/>
  </owl:Restriction>
</owl:equivalentClass>
</owl:Class>
```

2.4. Γράφοι απεικόνισης οντολογιών

Όπως έχει αναφερθεί και στην εισαγωγή, η διπλωματική εργασία αυτή πραγματεύεται την δημιουργία ενός εργαλείου οπτικοποίησης κατατμημένων οντολογιών. Σε αυτό το σημείο χρειάζεται να διαλευκανθούν δυο έννοιες, η οπτικοποίηση οντολογιών και η οπτικοποίηση κατατμημένων οντολογιών.

Ο τρόπος με τον οποίο το εργαλείο της διπλωματικής εργασίας οπτικοποιεί μια οντολογία είναι μέσω της δημιουργίας γράφων αναπαράστασης των στοιχείων αυτής. Ο γράφος αναπαράστασης οντολογιών είναι ένας γράφος που αποτελείται από κόμβους και ακμές. Οι κόμβοι αναπαριστούν τα στοιχεία κλάσεων μιας οντολογίας και οι ακμές που συνδέουν τους κόμβους αναπαριστούν τις σχέσεις μεταξύ αυτών των κλάσεων. Παρακάτω δίδεται ένα κομμάτι κώδικα μιας οντολογίας OWL-RDF/XML και το πώς μπορεί να οπτικοποιηθεί η πληροφορία που περιέχει.

```
<owl:Class rdf:ID="#man">
<rdfs:subClassOf rdf:resource="#human"/>
</owl:Class>
```



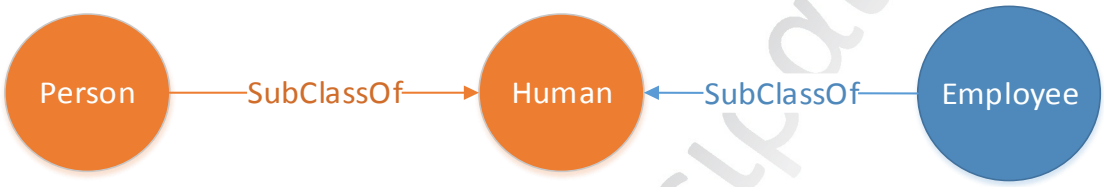
Εικόνα 5: Γραφική απεικόνιση μιας σχέσης υποκλάσεων OWL.

Όπως φαίνεται και στην Εικόνα 5 παρουσιάζεται η πληροφορία που υπάρχει στο κώδικα OWL, δηλαδή ότι η κλάση “man” είναι υποκλάση (SubClassOf) της κλάσης “human”. Η κεντρική ιδέα είναι ότι ο γράφος δεν οπτικοποιείται απευθείας κάνοντας συντακτική ανάλυση του OWL κώδικα. Αρχικά η συντακτική ανάλυση παράγει τριπλέτες στοιχείων της οντολογίας. Η κάθε μια τριπλέτα περιέχει τρία στοιχεία, δύο κλάσεις και την σχέση που υπάρχει μεταξύ τους (όταν πρόκειται για συσχέτιση κλάσεων). Δημιουργείται έτσι μια πρώιμη μορφή του γράφου, όπου αποτελείται από όλες τις τριπλέτες που περιγράφουν τα στοιχεία της οντολογίας και τις σχέσεις μεταξύ τους. Για την παρουσίαση ιδιοτήτων και περιορισμών σε αυτές χρησιμοποιείται άλλο πρότυπο που θα παρουσιαστεί παρακάτω.

Στην αρχική παράγραφο του κεφαλαίου αναφέρθηκε και η έννοια κατάτμηση οντολογιών. Ο τρόπος με τον οποίο μπορεί να απεικονιστεί η κατάτμηση οντολογιών είναι και αυτός μέσω της δημιουργίας γράφων. Αυτοί οι γράφοι πρέπει όμως να περιέχουν επιπρόσθετη πληροφορία ώστε να είναι διακριτό το ότι οι κόμβοι και οι ακμές μπορεί να αφορούν

στοιχεία διαφορετικών οντολογιών (π.χ. οι κόμβοι και οι ακμές που οπτικοποιούν σχέσεις διαφορετικών οντολογιών να έχουν διαφορετικό χρώμα, ή/και να βρίσκονται σε διαφορετικό χώρο επισκόπησης). Παρακάτω δίδεται ένα παράδειγμα όπου στο προηγούμενο γράφο απεικονίζεται και ένα στοιχείο μιας διαφορετικής οντολογίας(μέσω C-OWL) που έχει μια σχέση με ένα στοιχείο της αρχικής οντολογίας.

```
<cowl:bridgeRule>
 <cowl:Onto><cowl:source>
  <owl:Class rdf:about=" #employee"/>
 </cowl:source><cowl:target>
  <owl:Class rdf:about="http://localhost/figure4/unit1.owl#Human"/>
 </cowl:target></cowl:Onto>
</cowl:bridgeRule>
```



Εικόνα 6: Γραφική απεικόνιση μιας σχέσης υποκλάσεων OWL και μιας σχέσης COWL.

Στην Εικόνα 6 φαίνεται το πώς απεικονίζεται η σύνδεση του στοιχείου μιας δεύτερης οντολογίας με ένα στοιχείο της αρχικής. Η κατεύθυνση του μπλε βέλους δείχνει την φορά της σύνδεσης. Πιο συγκεκριμένα δείχνει ποια οντολογία θεωρεί ότι έχει σχέση με κάποια κλάση κάποιας άλλης οντολογίας. Στην προκειμένη περίπτωση παρουσιάζεται ότι η οντολογία που περιέχει την κλάση Employee θεωρεί ότι αυτή της η κλάση συνδέεται με την οντολογία που περιέχει την κλάση Human.

3. Στόχοι και απαιτήσεις εργαλείου ΕΟΚΟ

Σε αυτό το κεφάλαιο θα αναλυθούν οι στόχοι της παρούσας διπλωματικής εργασίας, καθώς και οι απαιτήσεις που γεννιούνται για την επίτευξη των στόχων αυτών. Πιο συγκεκριμένα, θα παρατεθούν όλα εκείνα τα λειτουργικά χαρακτηριστικά που καλείται να ενσωματώσει το Εργαλείο Οπτικοποίησης Κατατμημένων Οντολογιών, ώστε να επιτύχει τους στόχους του, αλλά και να διαφοροποιηθεί από τα υπάρχοντα εργαλεία οπτικοποίησης οντολογιών.

3.1. Στόχοι του ΕΟΚΟ

Υπάρχουν αρκετά αξιόλογα εργαλεία για την οπτικοποίηση οντολογιών, το καθένα έχει δημιουργηθεί για να πετύχει κάποιο συγκεκριμένο στόχο. Οι στόχοι που καλείται να επιτύχει το εργαλείο ΕΟΚΟ είναι οι εξής:

- Δημιουργία γράφων οπτικοποίησης οντολογιών.
- Δημιουργία ενός διαδραστικού εργαλείου απεικόνισης οντολογιών.
- Εκτεταμένη οπτικοποίηση οντολογιών για μεγαλύτερου βάθους μελέτη.
- Δημιουργία γράφων που απεικονίζουν τις σχέσεις μεταξύ διαφορετικών οντολογιών.

Ο όρος «γράφος απεικόνισης και μελέτης οντολογιών» υποδηλώνει την δημιουργία ενός γράφου που θα απεικονίζει όλες τις κλάσεις και τις σχέσεις μίας οντολογίας, ώστε να είναι δυνατή η μελέτη της με πιο αποδοτικό και εύκολο τρόπο σε σχέση με την μελέτη ενός OWL-RDF/XML αρχείου. Η οπτικοποίηση μιας οντολογίας κάνει πολύ πιο ευδιάκριτη όλη τη πληροφορία που περιέχει. Η απλή οπτικοποίηση μιας οντολογίας δεν είναι πολλές φορές αρκετή, διότι μια οντολογία μπορεί να είναι πολύ σύνθετη και να περιέχει μεγάλο όγκο πληροφορίας. Από μια τέτοια οντολογία θα δημιουργηθεί και ένας πολύ σύνθετος γράφος με πολλούς κόμβους και ακμές (κλάσεις και σχέσεις). Ένας σύνθετος γράφος χρειάζεται να γίνει πιο διαδραστικός, ώστε να μπορέσει να μελετηθεί επαρκώς στην όλη λεπτομέρεια που περιέχει. Πιο συγκεκριμένα, χρειάζεται να υπάρχει η δυνατότητα τροποποίησής του (μεγέθυνση, σμίκρυνση, μετακίνηση κόμβων, αναζήτηση κόμβων).

Μία οντολογία μπορεί να μελετηθεί αρχικά βάσει μιας βασικής απεικόνισης που αποτελείται από τις κλάσεις και τις μεταξύ τους σχέσεις υποκλάσεων. Αυτή η βασική απεικόνιση δίνει μια γενικότερη εικόνα για τη δομή μιας οντολογίας. Ωστόσο υπάρχει πολύ περισσότερη πληροφορία (σχέσεις μεταξύ κλάσεων, ιδιότητες, περιορισμοί) μέσα σε μια οντολογία, που είναι εξίσου σημαντική. Αυτή η πληροφορία για να απεικονιστεί χρειάζεται τη δημιουργία ενός γράφου που θα απεικονίζει σε μεγαλύτερο βάθος και πυκνότητα την οντολογία (περισσότεροι κόμβοι και σχέσεις). Ένας γράφος που θα περιέχει όλη τη πληροφορία που υπάρχει μέσα σε μια οντολογία -ακόμα και αν αφορά μια μικρή οντολογία- μπορεί να γίνει πολύ χαοτικός και πολύπλοκος. Λόγω αυτού του φαινομένου για να επιτευχθεί εις βάθος μελέτη και ανάλυση της οντολογίας, απαιτείται η προσθήκη επιπλέον λειτουργικότητας και διαδραστικότητας (Κεφάλαιο 3.2 επιπλέον λειτουργικά χαρακτηριστικά).

Όπως αναφέρθηκε και στην αρχή του κεφαλαίου, το ΕΟΚΟ έχει ως στόχο τη δημιουργία γράφου που να απεικονίζονται οι σχέσεις μεταξύ οντολογιών. Ο συγκεκριμένος στόχος αποτελεί και το κύριο στόχο και σκοπό δημιουργίας του εργαλείου ΕΟΚΟ. Κανένα από τα ήδη υπάρχοντα εργαλεία (Κεφάλαιο 7.1) δεν δίνει τη δυνατότητα της απεικόνισης σχέσεων

μεταξύ οντολογιών (C-OWL σχέσεων). Το ΕΟΚΟ θα παρουσιάζει στο γράφο απεικόνισης της οντολογίας και σχέσεις με διαφορετικές οντολογίες. Πιο συγκεκριμένα θα απεικονίζει και πληροφορίες που υπάρχουν σε αρχεία C-OWL και αφορούν τις συσχετίσεις μεταξύ οντολογιών που έχουν απεικονιστεί.

3.2. Απαιτήσεις υλοποίησης του ΕΟΚΟ

Οι στόχοι που αναφερθήκαν στο προηγούμενο κεφάλαιο γεννούν τις απαιτήσεις για την υλοποίηση του εργαλείου ΕΟΚΟ. Αρχικά αναφέρθηκε ότι το ΕΟΚΟ καλείται να δημιουργήσει γράφους οπτικοποίησης οντολογιών. Αυτός ο στόχος δημιουργεί συγκεκριμένες απαιτήσεις. Το ΕΟΚΟ πρέπει να επεξεργάζεται OWL-RDF/XML αρχεία ώστε να μπορέσει να δημιουργήσει το γράφο μιας οντολογίας. Πιο συγκεκριμένα υπάρχει η απαίτηση το ΕΟΚΟ να αναλύει τα OWL αρχεία και να μεταφράζει την πληροφορία που περιέχουν σε ένα γράφο. Αυτός ο γράφος δεν είναι αναγκαίο να παρουσιάζει σε πρώτο επίπεδο όλη την πληροφορία της OWL οντολογίας· η απεικόνιση της βασικής δομής της είναι αρκετή για να ξεκινήσει κάποιος να μελετάει μια (κατατμημένη) οντολογία.

Στην συνέχεια αναφέρθηκε ότι το ΕΟΚΟ έχει στόχο τη δημιουργία ενός διαδραστικού εργαλείου. Οι απαιτήσεις που γεννιούνται από αυτόν το στόχο αφορούν κυρίως την διεπαφή του εργαλείου. Για να επιτευχθεί η διαδραστικότητα, το εργαλείο πρέπει να έχει τα παρακάτω χαρακτηριστικά:

- Δυνατότητα σμίκρυνσης και μεγέθυνσης γράφου.
- Δυνατότητα επιλογής ενός ή περισσότερων κόμβων και μετακίνηση τους.
- Δυνατότητα αναζήτησης κόμβων.
- Δυνατότητα απεικόνισης πολλαπλών οντολογιών και περιήγησης μεταξύ αυτών.

Ένας σημαντικός στόχος του ΕΟΚΟ είναι η εις βάθος μελέτη της οντολογίας. Για να μπορέσει μια οντολογία να μελετηθεί εις βάθος, χρειάζεται στο γράφο οπτικοποίησης της να υπάρχει όλη η πληροφορία που περιέχει. Όπως αναφέρθηκε, όμως, και στο προηγούμενο κεφάλαιο η απεικόνιση όλης της πληροφορίας στον αρχικό γράφο μπορεί εύκολα να δημιουργήσει έναν πολύ περίπλοκο και δυσανάγνωστο γράφο. Εμφανίζεται έτσι η απαίτηση της δημιουργίας ενός επιπρόσθετου γράφου που θα αφορά τις σύνθετες σχέσεις που υπάρχουν σε μια οντολογία. Πιο συγκεκριμένα στο χρήστη χρειάζεται να δίνεται η επιλογή να επιλέξει κάποιο κόμβο (κλάση της οντολογίας) και να του εμφανίζεται ένας γράφος με όλη την επιπρόσθετη πληροφορία που τον αφορά (ρόλοι, περιορισμοί, ιδιότητες). Αυτό το χαρακτηριστικό θα βοηθήσει επίσης και ως προς την λειτουργικότητα, καθώς θα διευκολύνει τον χρήστη στην επιμέρους και σε βάθος ανάλυση και μελέτη των οντολογιών χωρίς να περιπλέκει πολλαπλούς γράφους.

Η σημαντικότερη απαίτηση για τη δημιουργία του ΕΟΚΟ είναι η απεικόνιση C-OWL σχέσεων στους γράφους των οντολογιών. Για να υλοποιηθεί αυτή η λειτουργία είναι απαραίτητη η επεξεργασία και ανάλυση των C-OWL αρχείων και η ενσωμάτωση των σχέσεων που δηλώνουν στους γράφους των οντολογιών. Για να μπορέσει να διατηρηθεί η λειτουργικότητα και να μην γίνουν πολύπλοκοι οι γράφοι, η πληροφορία που αφορά στις C-OWL σχέσεις πρέπει να παρουσιάζεται και αυτή στους επιμέρους γράφους των κλάσεων που αναφέρθηκαν στην προηγούμενη παράγραφο. Οι C-OWL σχέσεις αφορούν διαφορετικές οντολογίες μεταξύ τους. Λόγω αυτού του γεγονότος είναι σημαντικό ο χρήστης να μπορεί να έχει στην διάθεσή του και τις απεικονίσεις όλων των εμπλεκομένων

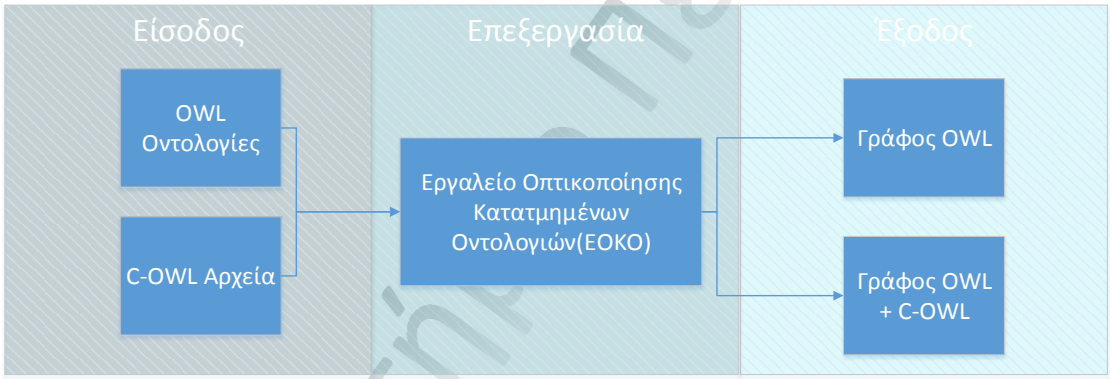
οντολογιών, παράλληλα. Με αυτό τον τρόπο θα μπορεί να μελετήσει και την δομή των οντολογιών που σχετίζονται με την αρχική οντολογία. Για να επιτευχθεί συνεπώς αυτός ο σκοπός, καθώς και για να παραμείνει λειτουργικό το εργαλείο, είναι αναγκαία η απεικόνιση αλλά και η δυνατότητα πλοήγησης μεταξύ πολλαπλών οντολογιών και συσχετίσεων.

4. Περιγραφή Εργαλείου Οπτικοποίησης Κατατμημένων οντολογιών (EOKO)

Το Εργαλείο Οπτικοποίησης Κατατμημένων Οντολογιών (EOKO) αποτελεί ένα σύστημα που δημιουργεί γράφους αναπαράστασης οντολογιών σχεδιασμένες σε γλώσσα OWL-DL με εκφραστικότητα το πολύ SHIQ και παράλληλα δημιουργεί γράφους εξάρτησης μεταξύ αυτών των οντολογιών κάνοντας ανάλυση C-OWL αρχείων.

4.1. Περιγραφή σχεδιασμού εργαλείου

Το EOKO έχει δυο κύρια χαρακτηριστικά, την γραφική απεικόνιση οντολογιών και την ενσωμάτωση σε αυτή τη γραφική αναπαράσταση των συσχετίσεων μεταξύ αυτών των οντολογιών (Εικόνα 7). Έχει προγραμματιστεί σε γλώσσα Java σε περιβάλλον Eclipse. Πιο συγκεκριμένα, το πρόγραμμα δέχεται ως είσοδο μία έως τέσσερις οντολογίες και δημιουργεί το γράφο των σχέσεων μεταξύ στοιχείων κάθε οντολογίας (οι λόγοι που μπορεί να δεχθεί μέχρι τέσσερεις οντολογίες αναφέρονται στο κεφάλαιο «Συμπεράσματα προγραμματιστικών επιλογών και υλοποίησης του EOKO»). Παράλληλα, το πρόγραμμα δέχεται και ως είσοδο ένα αρχείο C-OWL που περιέχει όλες τις σχέσεις μεταξύ των οντολογιών, αναλύει αυτό το αρχείο και απεικονίζει την επιπρόσθετη πληροφορία.



Εικόνα 7: Απεικόνιση εισόδων και εξόδων του Εργαλείου Οπτικοποίησης Κατατμημένων οντολογιών.

4.2. Αρχιτεκτονική εργαλείου

Σε αυτό το κεφάλαιο θα αναλυθεί η αρχιτεκτονική και θα παρουσιαστεί η υλοποίηση του εργαλείου EOKO. Πιο συγκεκριμένα θα παρουσιαστεί του πως το EOKO μετατρέπει μια οντολογία (ένα αρχείο OWL-RDF/XML) σε ένα γράφο απεικόνισης όπου απεικονίζονται όλα τα στοιχεία της οντολογίας, καθώς και το πώς προσθέτει επιπλέον πληροφορία που αφορά στις σχέσεις μεταξύ στοιχείων διαφορετικών οντολογιών (C-OWL συνδέσεων). Αρχικά θα γίνει μια βασική παρουσίαση των τριών κύριων σταδίων-λειτουργικών μονάδων υλοποίησης του EOKO (axiomanal, axiomtostring, Graph_Creator) και στη συνέχεια θα γίνει εκτεταμένη ανάλυση αυτών των λειτουργικών μονάδων (πώς λειτουργούν και αλληλοεπιδρούν).

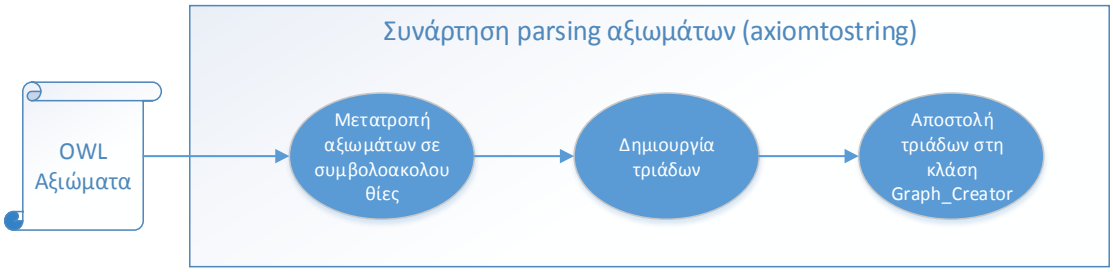
4.2.1. Βασική αρχιτεκτονική

Το EOKO αποτελείται από τρείς κύριες λειτουργικές μονάδες. Η πρώτη λειτουργική μονάδα είναι εκείνη που αναλύει συντακτικά τα αρχεία OWL-DL (axiomanal). Πιο συγκεκριμένα, η μονάδα αυτή υλοποιείται από μια συνάρτηση, που αναλύει χρησιμοποιώντας τη βιβλιοθήκη OWL-API, την οντολογία, ώστε να εξαγάγει από αυτή όλα τα λογικά αξιώματα που την δομούν. Η εξαγωγή όλων των λογικών αξιωμάτων είναι

μία διαδικασία σημαντική, διότι μέσα σε αυτά υπάρχει όλη η πληροφορία για τις συνδέσεις μεταξύ των στοιχείων της οντολογίας. Η λειτουργική μονάδα αυτή (axiomanal) επίσης αποδομεί και τα αρχεία C-OWL. Λόγω της φύσης όμως της πληροφορίας μέσα σε ένα πλαίσιο C-OWL δεν είναι δυνατό να εξαχθούν αξιώματα, παρά μόνο να αναλυθεί το αρχείο χρησιμοποιώντας τη βιβλιοθήκη Dom4j που κάνει ανάλυση XML αρχείων, ώστε να εξαχθεί η απαραίτητη πληροφορία για την δημιουργία των C-OWL συνδέσεων μεταξύ των στοιχείων της οντολογίας. Στη συνέχεια η πρώτη λειτουργική μονάδα (axiomanal) στέλνει κάθε αξίωμα της οντολογίας στη δεύτερη λειτουργική μονάδα (axiomtostring), ώστε να το επεξεργαστεί σε μεγαλύτερο βάθος. Στην Εικόνα 8 παρατίθεται η αλληλουχία αυτών των διαδικασιών. Στο πάνω μέρος της Εικόνας 8 φαίνεται η διαδικασία ανάλυσης C-OWL αρχείων και στο κάτω μέρος των OWL οντολογιών.

Εικόνα 8 :Λειτουργία συνάρτησης ανάλυσης αξιωμάτων(axiomanal).

Ο λόγος ύπαρξης της δεύτερης λειτουργικής μονάδας (axiomtostring) είναι ότι ένα αξίωμα μπορεί να περιέχει απλές έννοιες, δηλαδή να περιέχει δυο κλάσεις και τη σχέση μεταξύ αυτών (π.χ. C1⊑C2 (η κλάση C1 είναι υποκλάση της C2)), αλλά μπορεί να είναι ένα γενικευμένο αξίωμα που περιέχει σύνθετες έννοιες, και συνεπώς τα μέρη του αξιώματος μπορεί να δομούνται από απλούστερες έννοιες (π.χ. A⊑(C_1 ⊓C_2)). Τα γενικευμένα αξιώματα δημιουργούν την ανάγκη για περεταίρω επεξεργασία ώστε να αναδειχθεί ο τρόπος συσχέτισης διαφορετικών εννοιών. Πιο συγκεκριμένα, απαιτείται μια αναδρομική μέθοδος για την ανάλυση των εννοιών μέχρι τις απλούστερες δυνατές (ονόματα κλάσεων). Μόλις γίνει αυτή η επεξεργασία ακολουθεί η δημιουργία των συνδέσεων, ώστε να τροφοδοτηθούν στη τρίτη λειτουργική μονάδα (Graph_creator) που αναλαμβάνει την απεικόνιση των οντολογιών. Η δημιουργία των συνδέσεων είναι η μετατροπή της πληροφορίας σε μορφή που να μπορεί να επεξεργαστεί από την τρίτη λειτουργική μονάδα που δημιουργεί το γράφο. Οι συνδέσεις μετατρέπονται σε τριάδες στοιχείων (τριπλέτες). Οι τριπλέτες αυτές περιέχουν δύο κλάσεις της οντολογίας που πλέον μεταφράζονται σαν κόμβοι του γράφου και το τρίτο στοιχείο της τριάδας είναι η σχέση μεταξύ των δύο κλάσεων που πλέον μεταφράζεται (απεικονίζεται) ως η ακμή που συνδέει τους δύο κόμβους-κλάσεις . Στην Εικόνα 9 παρουσιάζεται η διαδικασία παραγωγής τριάδων και η αποστολή τους μετέπειτα στην κλάση Graph_Creator.

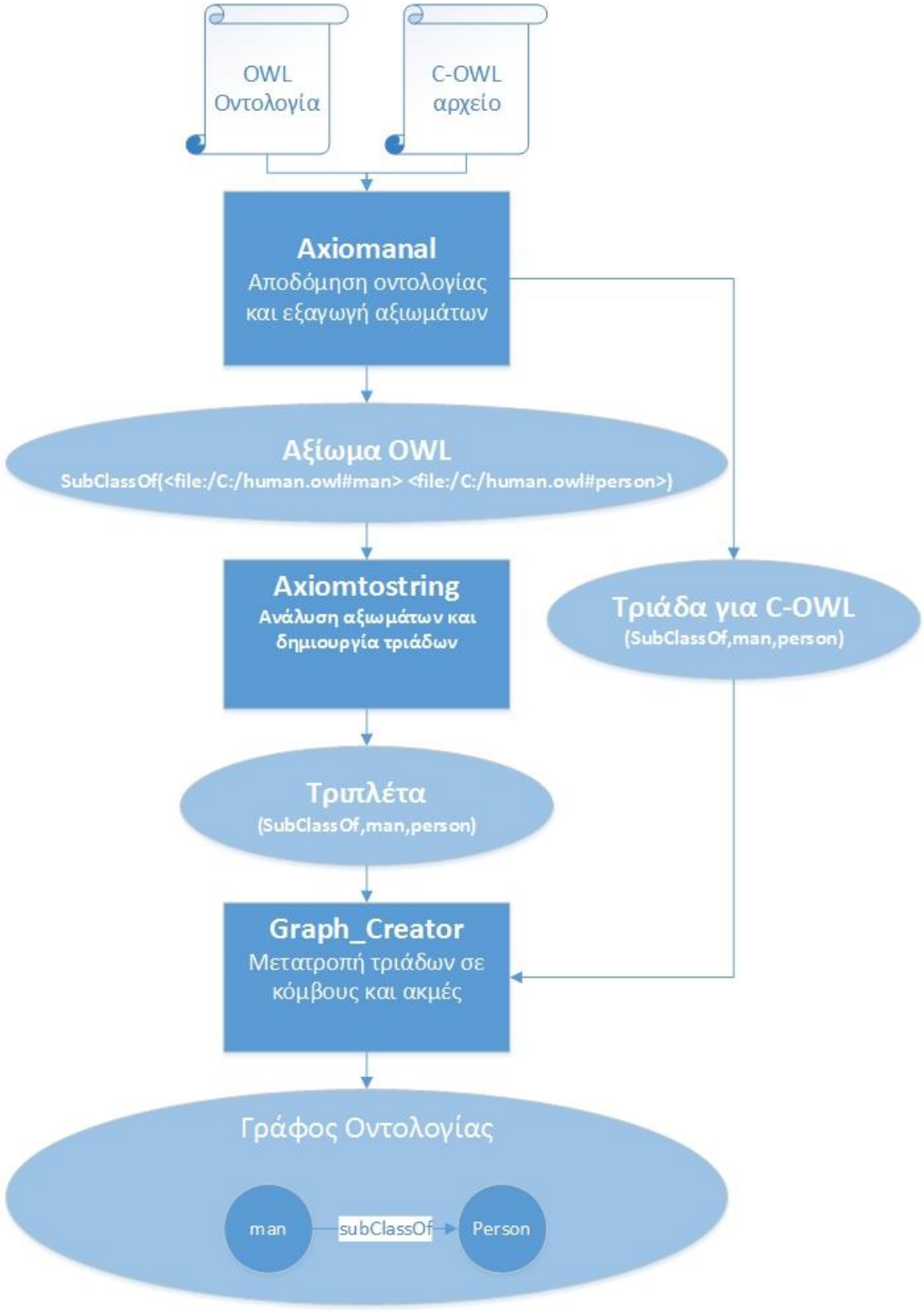


Εικόνα 9: Λειτουργία συνάρτησης parsing αξιωμάτων(axiomtostring).

Η τρίτη λειτουργική μονάδα δημιουργεί τον γράφο της οντολογίας (Graph_creator). Τροφοδοτείται με τις τριάδες που δημιουργήθηκαν από την δεύτερη μονάδα και χρησιμοποιώντας τη βιβλιοθήκη Jung της Java δημιουργεί το γράφο της οντολογίας. Επίσης, προσθέτει επιπλέον λειτουργικότητα στο γράφο δίνοντας τη δυνατότητα για μετακίνηση των κόμβων, σμίκρυνση και μεγέθυνση του γράφου, εναλλαγή μεταξύ οντολογιών που έχουν απεικονιστεί, καθώς και αναζήτηση κόμβων (Εικόνα 10).

Εικόνα 10:Λειτουργεία κλάσης δημιουργίας γράφου (graph_creator).

Η σύνδεση και η επικοινωνία των τριών λειτουργικών μονάδων που συνθέτουν το πρόγραμμα (axiomanal, axiomtostring, graph_creator), φαίνεται στην Εικόνα 11. Πιο συγκεκριμένα στην Εικόνα 11 παρουσιάζεται το πώς κάθε λειτουργική μονάδα λαμβάνει και επεξεργάζεται την πληροφορία και την μεταφέρει στην επόμενη λειτουργική μονάδα, ώστε να δημιουργηθεί ο αντίστοιχος γράφος απεικόνισης.

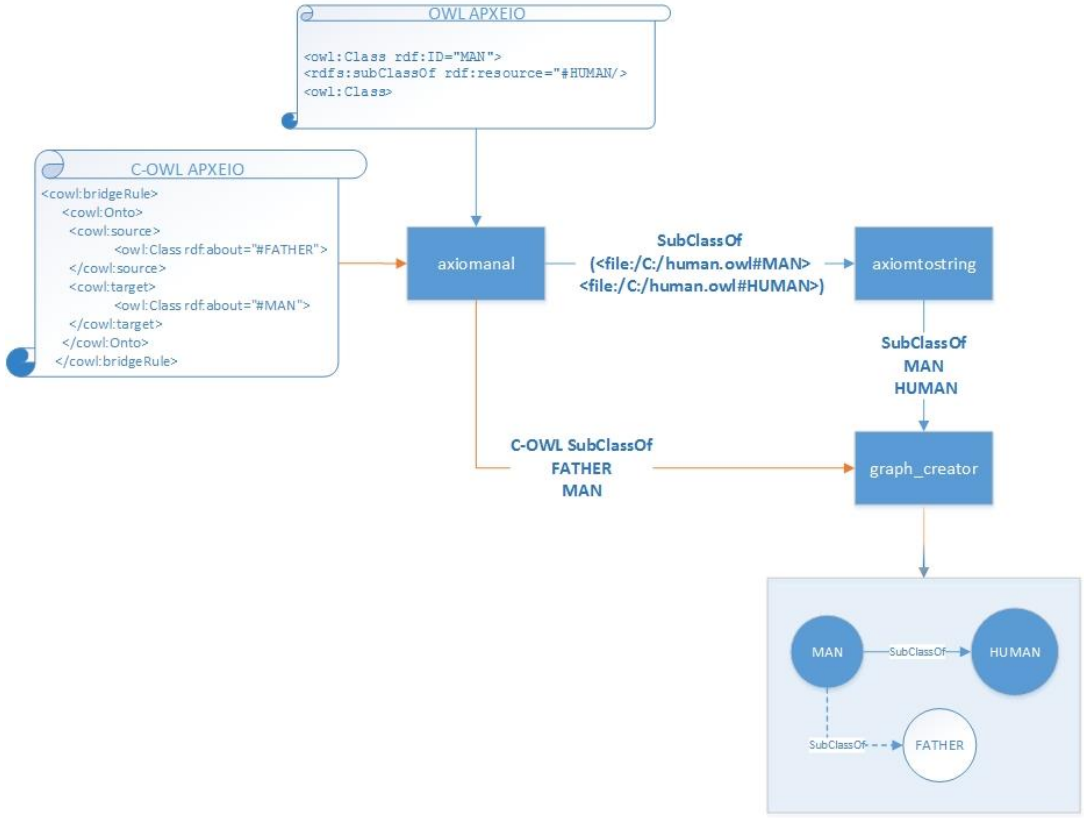


Εικόνα 11: Βασική αρχιτεκτονική Εργαλείου Οπτικοποίησης Κατατμημένων οντολογιών (ΕΟΚΟ).

4.2.2. Εκτεταμένη ανάλυση υλοποίησης του ΕΟΚΟ

Έχοντας παρουσιάσει τη βασική αρχιτεκτονική του ΕΟΚΟ, παρατίθεται η αναλυτική αρχιτεκτονική υλοποίησης της καθεμίας από τις λειτουργικές μονάδες του (axiomanal, axiomtostring, graph_creator). Οι τρείς λειτουργικές μονάδες αυτές στοχεύουν στην

μετατροπή της πληροφορίας, που υπάρχει μέσα σε μια οντολογία, σε ένα γράφο απεικόνισης στο μεγαλύτερο δυνατόν επίπεδο λεπτομέρειας της πληροφορίας, και με λειτουργικότητα διάδρασης. Όπως φαίνεται και στην Εικόνα 12, αρχικά από την OWL οντολογία εξάγονται τα OWL αξιώματα. Η εξαγωγή των αξιωμάτων από την οντολογίας υποδηλώνει ότι οι σχέσεις που υπάρχουν μέσα σε μια οντολογία γράφονται με έναν πιο απλό και εύκολο να επεξεργαστεί τρόπο(σαν αξιώματα). Στη συνέχεια τα αξιώματα μετατρέπονται σε συμβολοακολουθίες, οι συμβολοακολουθίες μετατρέπονται σε τριάδες και οι τριάδες σε κόμβους και ακμές του γράφου (Εικόνα 12).



Εικόνα 12: Μετατροπή πληροφορίας από την κάθε λειτουργική μονάδα του ΕΟΚΟ.

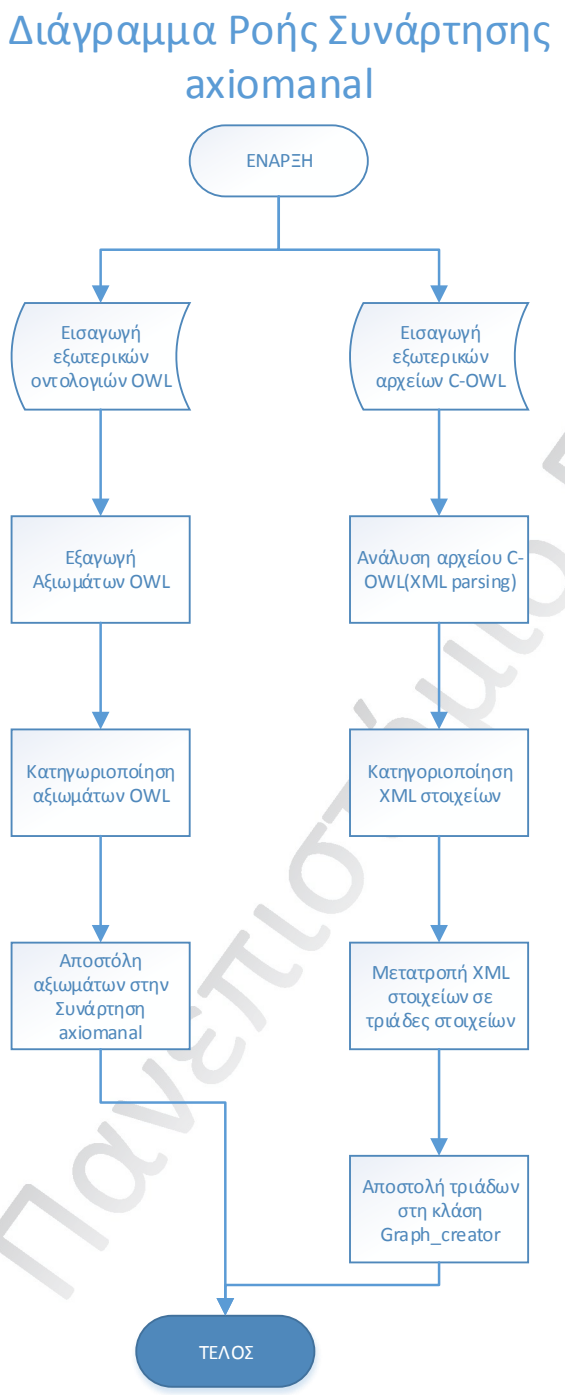
4.2.2.1. Συνάρτηση ανάλυσης αξιωμάτων (axiomanal)

Η συνάρτηση ανάλυσης αξιωμάτων έχει διπλή λειτουργία, αναλύει την κατατμημένη οντολογία, καθώς και τα C-OWL αρχεία (Εικόνα 12).

Όσον αφορά την πρώτη λειτουργία, η συνάρτηση δέχεται σαν είσοδο ένα αρχείο OWL για κάθε επιμέρους οντολογία και χρησιμοποιώντας τη μέθοδο της βιβλιοθήκης OWL-API *getLogicalAxioms()*, αντλεί από αυτή όλα τα λογικά αξιώματα. Στην συνέχεια κατηγοριοποιεί τα αξιώματα (αν είναι subClassOf, EquivalentClass, DisjointClass κτλ.). Μόλις γίνει η κατηγοριοποίηση των αξιωμάτων στέλνονται στην συνάρτηση axiomtostring για επιπλέον ανάλυση, καθώς η πληροφορία που περιέχουν δεν είναι σε μορφή συμβατή για τη μεταφορά τους στη κλάση δημιουργίας του γράφου.

Στο δεύτερο σκέλος της συνάρτησης, όπως προαναφέρθηκε, γίνεται η ανάλυση του αρχείου C-OWL. Ειδικότερα, η συνάρτηση δέχεται ένα αρχείο C-OWL και κάνει πλήρη ανάλυση του περιεχομένου χρησιμοποιώντας την βιβλιοθήκη της Java Dom4j για ανάλυση

XML αρχείων. Όπως έχει αναφερθεί και στο κεφάλαιο «Προαπαιτούμενες γνώσεις», τα αρχεία OWL και κατ' επέκταση τα αρχεία C-OWL χρησιμοποιούν το συντακτικό των XML Schemas. Δοθέντος του ότι δεν υπάρχει επίσημη βιβλιοθήκη για την ανάλυση C-OWL αρχείων, ο μόνος τρόπος για τη ανάλυση τους είναι μέσω του "XML-Based" συντακτικού τους. Ο XML parser αναλύει το αρχείο C-OWL και μετατρέπει τη πληροφορία σε μορφή συμβατή για τη κλάση δημιουργίας του γράφου. Έπειτα αποστέλλει την πληροφορία αυτή στη κλάση graph_creator για την απεικόνισή της (Εικόνα 13).

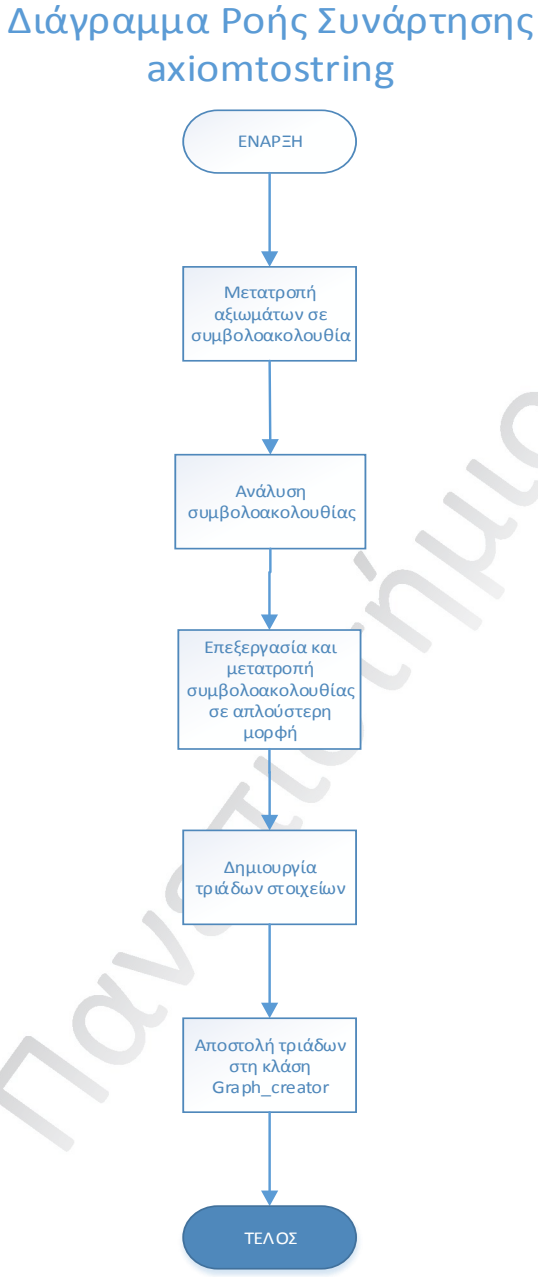


Εικόνα 13: Διάγραμμα ροής συνάρτησης axiomanal.

4.2.2.2. Συνάρτηση Parsing αξιωμάτων (axiomtostring)

Η συνάρτηση αυτή έχει ως σκοπό την μετατροπή της πληροφορίας σε μορφή που μπορεί να επεξεργαστεί η κλάση graph_creator. Πιο συγκεκριμένα, τα αξιώματα πρέπει να μετατραπούν σε τριάδες στοιχείων, κάθε τριάδα αποτελεί μια σύνδεση στο γράφο και έχει αύτη τη μορφή *τριάδα(ακμή, κόμβος1, κόμβος2)*. Η ακμή υποδηλώνει μια σχέση μεταξύ κλάσεων OWL και οι κόμβοι αποτελούν κλάσεις της OWL που αποτυπώνονται σαν κόμβοι του γράφου.

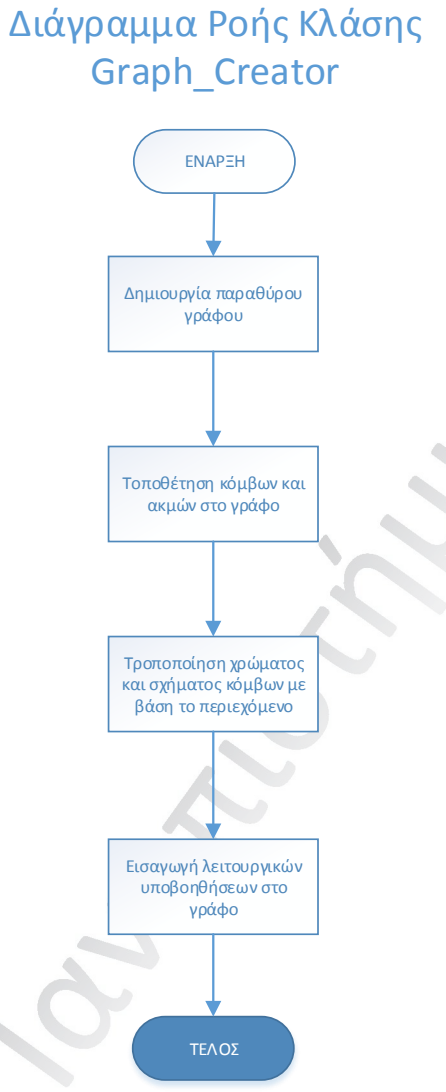
Η ανάλυση των αξιωμάτων μετατρέπει σε συμβολοακολουθίες και αντλεί τις κλάσεις και τις σχέσεις μέσα από αυτές με βάση τη σύνταξη των αξιωμάτων. Στη συνέχεια δημιουργεί τις τριάδες και τις μεταφέρει στη κλάση graph_creator ώστε να απεικονιστούν (Εικόνα 14).



Εικόνα 14: Διάγραμμα ροής συνάρτησης axiomtostring.

4.2.2.3. Κλάση δημιουργίας γράφου (graph_creator)

Η κλάση graph_creator δέχεται τις τριάδες στοιχείων C-OWL και OWL από τις συναρτήσεις axiomanal και axiomtostring αντίστοιχα και δημιουργεί το γράφο βάσει αυτών. Η κλάση αυτή πρώτα απεικονίζει τους γράφους και στη συνέχεια προσθέτει επιπρόσθετη πληροφορία πάνω σε αυτούς, με βάση τα χαρακτηριστικά τους. Τροποποιεί τα χρώματα των ακμών και των κόμβων αναλόγως του τι απεικονίζουν (διαφοροποίηση χρώματος κάθε οντολογίας, διαφοροποίηση ακμών με βάση το τύπο σύνδεσης που απεικονίζουν). Περισσότερες πληροφορίες για την επιπρόσθετη πληροφορία που υπάρχει στους γράφους απεικόνισης, υπάρχουν στο κεφάλαιο «Περιγραφή λειτουργίας εργαλείου Κεφάλαιο 4.3». Μία επιπλέον λειτουργία της κλάσης είναι να προσθέτει λειτουργικότητα στο γράφο, όπως αναφέρθηκε και στο κεφάλαιο της «Αρχιτεκτονικής» (Εικόνα 15).



Εικόνα 15: Διάγραμμα ροής κλάσης graph_creator.

4.3. Περιγραφή λειτουργίας εργαλείου

Σε αυτό το κεφάλαιο θα περιγραφούν τα λειτουργικά χαρακτηριστικά του εργαλείου, καθώς και οι σχεδιαστικές επιλογές που συντέλεσαν στην επιλογή και υλοποίηση αυτών των λειτουργικών χαρακτηριστικών.

4.3.1. Αναπαράσταση βασικού γράφου

Το ΕΟΚΟ έχει την δυνατότητα αναπαράστασης μέχρι τεσσάρων οντολογιών ταυτόχρονα και ενσωμάτωση στο γράφο πληροφοριών ενός C-OWL αρχείου. Αρχικά, το πρόγραμμα δημιουργεί ένα παράθυρο, όπου απεικονίζει τις οντολογίες σε διαφορετικές καρτέλες τη κάθε μια. Κάθε οντολογία βρίσκεται σε μία καρτέλα και οι κόμβοι της κάθε οντολογίας έχουν διαφορετικό χρώμα, ώστε να ξεχωρίζει από τις άλλες. Είναι σημαντικό να σημειωθεί ότι στον αρχικό γράφο κάθε οντολογίας εμφανίζονται μόνο σχέσεις μεταξύ υποκλάσεων (SubClassOf, EquivalentClass, disjointClass) και όχι ιδιότητες ή αξιώματα με σύνθετες έννοιες. Ο λόγος είναι ότι αν στον αρχικό γράφο παρουσιαζόταν όλη η πληροφορία, θα γινόταν πολύ περίπλοκος και δυσανάγνωστος, λόγω της πληθώρας ακμών και κόμβων. Αντίθετα, ένας γράφος που παρουσιάζει τη βασική δομή της οντολογίας, περιέχει σημαντικότατη πληροφορία για τη γενικότερη διάρθρωσή της και αποτελεί ένα ισχυρό εφαλτήριο για περεταίρω ανάλυση και μελέτη. Για να παρουσιαστεί παρ' όλα αυτά όλη η απαραίτητη πληροφορία για την οντολογία, χωρίς όμως να περιπλέκεται ο γράφος, έγιναν κάποιες λειτουργικές ενισχύσεις σε αυτόν.

Όταν ο χρήστης τοποθετεί τον κέρσορα πάνω από ένα κόμβο (hover), εμφανίζεται ένα παράθυρο που αναγράφει όλες τις σχέσεις μέσα στην οντολογία που τον αφορούν ανεξαρτήτως του τύπου και της πολυπλοκότητας τους. Μια ανάλογη λειτουργία υπάρχει και στις ακμές του γράφου. Οι ακμές του γράφου ανάλογα με τον τύπο σύνδεσης που περιγράφουν, έχουν διαφορετικό χρώμα: γκρι για τις σχέσεις υποκλάσεων και σχέσεων υποκλάσεων μεταξύ σύνθετων κλάσεων, κόκκινο για τις σχέσεις ξένων κλάσεων και πράσινο για τις σχέσεις ισοδύναμων κλάσεων. Με αυτό τον τρόπο γίνεται πιο εύκολο για το χρήστη να εντοπίσει τον τύπο των σχέσεων. Παράλληλα, όταν ο χρήστης τοποθετεί τον κέρσορά του πάνω από μία ακμή, εμφανίζεται ένα παράθυρο που αναγράφει την πλήρη σχέση που περιγράφει αυτή η ακμή (τις κλάσεις που ενώνει και τον τύπο της σχέσης).

Ο αλγόριθμος για την διάταξη των κόμβων είναι επίσης μια σημαντική λειτουργικότητα, διότι όταν μια οντολογία είναι μεγάλη σε μέγεθος (περιέχει πολλές κλάσεις), ακόμα και ένας βασικός αφαιρετικός γράφος που δημιουργείται αρχικά μπορεί να είναι πολύ πυκνός και να μην απεικονίζει τις οντολογίες όπως θα έπρεπε. Για το λόγο αυτό, ο αρχικός γράφος χρησιμοποιεί για τη διάταξη κόμβων τον αλγόριθμο *DAGLayout* της βιβλιοθήκης Jung. Ο αλγόριθμος αυτός διατάσσει τους κόμβους ιεραρχικά, τοποθετώντας τους γονικούς κόμβους (υπερκλάσεις της οντολογίας) σε υψηλότερο επίπεδο και στη συνέχεια από κάτω τους κόμβους παιδιά (υποκλάσεις της οντολογίας). Με αυτό τον τρόπο, όσο μεγάλη και να είναι η οντολογία χωρίζεται σε διακριτά επίπεδα, δημιουργώντας καλύτερη και πιο ευδιάκριτη απεικόνιση. Ο χρήστης όμως μπορεί να επέμβει στη διευθέτηση των κόμβων και να τους μετακινήσει κατά την προτίμησή του.

4.3.2. Αναλυτικός γράφος κόμβων

Για να μπορέσουν να αναπαρασταθούν οι οντολογίες πλήρως δίνεται η δυνατότητα επιλογής των κόμβων από το χρήστη. Μόλις επιλέξει ο χρήστης κάποιον κόμβο δημιουργείται ένας καινούριος γράφος σε πρόσθετο παράθυρο, όπου περιέχει όλες τις σχέσεις που αφορούν αυτό το συγκεκριμένο κόμβο. Ο καινούριος γράφος δεν αναπαριστά μόνο σχέσεις κλάσεων, αλλά και ιδιότητες, περιορισμούς αυτών, καθώς και αξιώματα με σύνθετες έννοιες, ώστε να δημιουργηθεί μια πλήρης εικόνα για τις σχέσεις που έχει ο συγκεκριμένος κόμβος-κλάση στην οντολογία. Οι ιδιότητες αναπαρίστανται σε μορφή

ακμών μεταξύ των κόμβων και έχουν χρώμα γαλάζιο ώστε να ξεχωρίζουν από τους άλλους τύπους ακμών (σχέσεις υποκλάσεων, ξένων κλάσεων, όμοιων κλάσεων).

Ο αλγόριθμος *ISOMLayout* που χρησιμοποιείται για την απεικόνιση των στοιχείων του αναλυτικού γράφου έχει διαφορετική λογική από τον *DAGLayout*. Συγκεκριμένα, ο αλγόριθμος αυτός τοποθετεί στο κέντρο του γράφου το κόμβο που έχει επιλεχθεί από τον αρχικό γράφο και τοποθετεί γύρω του τους κόμβους που έχουν κάποιου είδους σχέση με αυτόν.

4.3.3. Αναπαράσταση C-OWL σχέσεων

Στο καινούριο γράφο αναπαρίστανται και οι σχέσεις που έχουν εξαχθεί από τα C-OWL πλαίσια, κάτι που έχει ως αποτέλεσμα την εμφάνιση κόμβων άλλων οντολογιών μέσα σε αυτόν. Η διαφοροποίηση χρώματος των οντολογιών (οι κόμβοι κάθε οντολογίας έχουν διαφορετικό χρώμα), που έχει εισαχθεί από το πρόγραμμα, καθώς και η διαφοροποίηση του χρώματος των ακμών (οι ακμές των C-OWL σχέσεων έχουν μαύρο χρώμα) κάνει τις σχέσεις αυτές να είναι εύκολα διακριτές μεταξύ τον σχέσεων OWL που υπάρχουν στο γράφο.

4.3.4. Επιπρόσθετη λειτουργικότητα

Στους γράφους που δημιουργεί το εργαλείο δίνεται η δυνατότητα επιλογής και μετακίνησης τον κόμβων μέσα στο παράθυρο. Ο χρήστης με αυτό τον τρόπο μπορεί να επικεντρωθεί σε ένα συγκεκριμένο κομμάτι της οντολογίας, ξεχωρίζοντας το από τα άλλα και γενικότερα επιτρέπει την αναδιάταξη του γράφου για τις ανάγκες του χρήστη. Επιπλέον, επιτρέπει την σμίκρυνση και μεγέθυνση του γράφου, καθώς και την επαναφορά του στην αρχική του κατάσταση σε περίπτωση που χρειάζεται ο χρήστης να αναιρέσει αλλαγές που έχει κάνει. Επιπρόσθετα, δίνεται η δυνατότητα αναζήτησης κόμβων του γράφου, δηλαδή ο χρήστης συμπληρώνει σε ένα πεδίο το κόμβο που αναζητά και αν ο κόμβος υπάρχει χρωματίζεται μπλε ώστε να είναι διακριτός από τους άλλους και να μπορεί να εντοπιστεί από το χρήστη.

5. Παρουσίαση υλοποίησης

Σε αυτό το κεφάλαιο θα αναλυθεί εκτενώς ο κώδικάς Java που υλοποιεί το EOKO. Πιο συγκεκριμένα, θα αναλυθούν όλες οι προγραμματιστικές επιλογές που έγιναν για την σωστή και πλήρη απεικόνιση μιας οντολογίας. Η έννοια σωστή απεικόνιση μεταφράζεται ως τη δημιουργία ενός γράφου που απεικονίζει πλήρως τη δομή της οντολογίας. Αναλυτικότερα, απεικονίζει όλες τις σχέσεις και τα αντικείμενα που υπάρχουν στην οντολογία (π.χ. στο βασικό γράφο να υπάρχουν όλες οι κλάσεις και οι σχέσεις υποκλάσεων, ξένων κλάσεων και ίσων κλάσεων).

Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο, το EOKO έχει προγραμματιστεί σε γλώσσα Java σε περιβάλλον Eclipse. Πέραν από τη βασική βιβλιοθήκη της Java JRE system library που χρησιμοποιείται για να μπορέσει να λειτουργήσει κάθε Java πρόγραμμα, το EOKO βασίστηκε σε τρεις κύριες βιβλιοθήκες ώστε να μπορέσει να υλοποιήσει το γράφο εξάρτησης. Αυτές οι βιβλιοθήκες είναι οι:

- OWL-API: Η διεπαφή αυτή χρησιμοποιήθηκε για να αναλύσει συντακτικά τις οντολογίες που εισήχθησαν από το χρήστη στο πρόγραμμα.
- Jung: Η βιβλιοθήκη αυτή χρησιμοποιήθηκε για να δημιουργήσει το γράφο κάθε οντολογίας.
- Dom4j: Η βιβλιοθήκη αυτή χρησιμοποιήθηκε για την ανάλυση των C-OWL αρχείων.

Η κλάση που υλοποιεί το πρόγραμμα είναι η *graph_creator*, μέσα στην οποία υπάρχουν όλες οι μέθοδοι και όλες οι συναρτήσεις που αναλύουν τα OWL και C-OWL αρχεία, καθώς και δημιουργούν τους γράφους. Αρχικά το πρόγραμμα καλεί τη μέθοδο κατασκευής των γράφων (*graph_creator)* των οντολογιών. Η μέθοδος αυτή παίρνει ένα όρισμα, το οποίο προσδιορίζει τον τύπο του γράφου που πρέπει να κατασκευάσει. Οι τύποι των γράφων είναι δύο, ο γενικός γράφος της οντολογίας που περιέχει μόνο την απεικόνιση της βασικής της δομής (κλάσεις, σχέσεις υποκλάσεων κ.ο.κ.) και ο γράφος απεικόνισης των σχέσεων μια συγκεκριμένης κλάσης που έχει επιλέξει ο χρήστης. Στην αρχική κλήση της μεθόδου *graph_creator* το όρισμα παίρνει τιμή ώστε να δημιουργήσει το γενικό γράφο κάθε οντολογίας που έχει εισαχθεί στο πρόγραμμα. Στη συνέχεια η μέθοδος *graph_creator*, αφού δημιουργήσει το αρχικό παράθυρο με βάση το αρχικό όρισμα (αν το όρισμα είναι για τη δημιουργία του βασικού γράφου δημιουργεί παράθυρο που απεικονίζει τους κόμβους σε διάταξη *DAGLayout*, αλλιώς δημιουργεί παράθυρο για διάταξη *ISOMLayout*), καλεί τη μέθοδο *axiomanal*, ώστε να αναλυθούν οι οντολογίες και να επιστραφούν οι κόμβοι και οι ακμές που πρόκειται να απεικονιστούν.

Η μέθοδος *axiomanal* δέχεται και αυτή ένα όρισμα, το ίδιο με την *graph_creator*. Στην ουσία μεταφέρεται το όρισμα από τη μια μέθοδο στην άλλη, διότι αφορά και τις δύο. Μέσα στην μέθοδο *axiomanal* χρησιμοποιούνται μέθοδοι από το OWL-API ώστε να γίνει η ανάλυση των αξιωμάτων. Πιο συγκεκριμένα, κάθε αρχείο OWL αποθηκεύεται σε μια μεταβλητή τύπου *OWLOntology* και στη συνέχεια εξάγονται από αυτή τη μεταβλητή όλα τα λογικά αξιώματα χρησιμοποιώντας τη μέθοδο *getLogicalAxioms()*. Στη συνέχεια τα λογικά αξιώματα ελέγχονται για τον τύπο τους και στέλνονται στη συνάρτηση *axiomtostring*.

Η συνάρτηση *axiomtostring* δέχεται σαν όρισμα το αξίωμα αφού μετατραπεί σε συμβολοακολουθία, και τη μεταβλητή που ορίζει τον τύπο του γράφου. Όντας πλέον το αξίωμα συμβολοακολουθία, μπορεί να υποστεί επεξεργασία για την εξαγωγή της πληροφορίας. Ένα αξίωμα που έχει εξαχθεί μέσω του OWL-API μπορεί να περιέχει σύνθετες η απλές έννοιες. Αναλόγως με τον τύπο γράφου που πρόκειται να απεικονιστεί, η συνάρτηση επιλέγει ποια αξιώματα πρόκειται να αναλύσει. Αυτό σημαίνει ότι αν έχει κληθεί η συνάρτηση για δημιουργία του γενικού γράφου, τότε η συνάρτηση θα αναλύσει μόνο τα αξιώματα με απλές έννοιες. Μόλις το πρόγραμμα εξάγει από το αξίωμα τις κλάσεις και τον τύπο σχέσεων, καλεί τη συνάρτηση *addEdge* που παίρνει τρία ορίσματα· ένα όρισμα είναι η ακμή και τα αλλά δυο είναι οι κόμβοι που θα συνδέει αυτή η ακμή. Στο πεδίο της ακμής τοποθετείται η σχέση που υπάρχει μεταξύ των δυο κλάσεων που υπάρχουν μέσα στο αξίωμα, ενώ στα πεδία των κόμβων μπαίνουν τα ονόματα των κλάσεων, ώστε να αναπαρασταθούν σαν κόμβοι στο γράφο. Η συνάρτηση στέλνει την πληροφορία αυτή στη μέθοδο *graph_creator* για απεικόνιση. Αυτή η διαδικασία ακολουθείται για όλα τα αξιώματα, ώστε να απεικονιστούν όλες οι σχέσεις.

Κατά το τέλος της κλήσης της συνάρτησης *axiomtostring* από την *axiomanal* ακολουθεί η διαδικασία για την ανάλυση των C-OWL αρχείων. Δυστυχώς δεν υπάρχει κάποιο API που να βοηθάει στην ανάλυση C-OWL αρχείων, όπως το OWL-API για την OWL. Ο μόνος τρόπος ανάλυσης των C-OWL αρχείων γίνεται μόνο μέσω XML ανάλυσης. Τα στοιχεία του αρχείου C-OWL μπορούν να αναγνωριστούν ως XML Elements και Attributes, επομένως, χρησιμοποιώντας την XML Parsing βιβλιοθήκη της Java Dom4j, μπορεί να γίνει ανάλυση των C-OWL αρχείων. Με βάση το συντακτικό των C-OWL αρχείων μπορούν να βγουν κανόνες για το ποια αντικείμενα μέσα στο αρχείο αυτό περιέχουν τη πληροφορία για την απεικόνιση των σχέσεων. Οι απλές σχέσεις υποκλάσεων είναι εύκολο να εξαχθούν, διότι έχουν απλή μορφή. Οι ιδιότητες και οι περιορισμοί σε αυτές είναι πιο σύνθετες, αλλά για τις ανάγκες του εργαλείου και της απεικόνισης δεν υπάρχει ανάλυση μεγάλου βάθους. Αναλυτικότερα, οι απλές σχέσεις υποκλάσεων ορίζονται μέσα σε ένα στοιχείο με όνομα *cowl:Into* και μέσα σε αυτό το στοιχείο υπάρχουν οι δυο κλάσεις που αφορά αυτή η σχέση μέσα σε στοιχεία με όνομα *owl:Class*. Έτσι μπορούν να εξαχθούν εύκολα και να τοποθετηθούν σαν ορίσματα μέσα στη μέθοδο *addEdge*, ώστε να απεικονιστούν στον αναλυτικό γράφο κάποιου κόμβου, εφόσον τον αφορούν.

Μόλις τελειώσει η κλήση της μεθόδου *axiomanal* η μέθοδος *graph_creator* έχει όλη την απαραίτητη πληροφορία για τη δημιουργία του γράφου. Απεικονίζει όλες τις σχέσεις που έχουν σταλεί σε αυτή μέσω της συνάρτησης *addEdge* και τροποποιεί το γράφο στη συνέχεια ανάλογα με το περιεχόμενο των κόμβων και των ακμών. Πιο συγκεκριμένα τοποθετεί κάθε οντολογία σε διαφορική καρτέλα μέσα στο παράθυρο και τη χρωματίζει διαφορετικά. Στη συνέχεια ελέγχει το τύπο των ακμών και τις χρωματίζει κατάλληλα.

Όταν ο χρήστης επιλέξει με το κέρσορα κάποιο κόμβο-κλάση ώστε να λάβει λεπτομερέστερη πληροφορία για τις ιδιότητες και τις σχέσεις τις, τότε το σύστημα ξανακαλεί τη συνάρτηση *graph_creator* που δημιουργεί ένα καινούριο παράθυρο, ώστε να απεικονίσει τις σχέσεις που αφορούν αυτό το κόμβο. Η διαδικασία για τη δημιουργία του γράφου είναι η ίδια με την αρχική, με τη διαφορά ότι εμφανίζονται όλοι οι τύποι σχέσεων πλέον, με μόνη εξαίρεση ότι η κάθε σχέση που θα απεικονιστεί, θα αφορά το κόμβο που έχει επιλεχθεί. Πιο συγκεκριμένα στο όρισμα που ορίζει τον τύπο γράφου

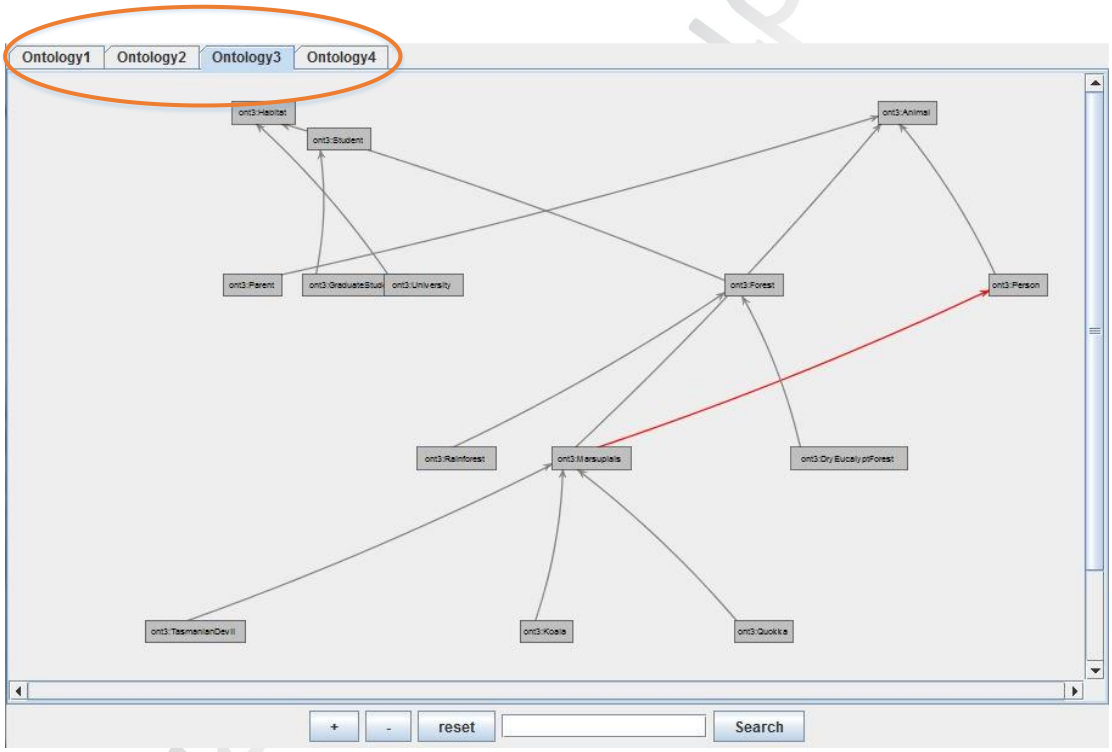
τοποθετείται το όνομα του κόμβου που επιλέχθηκε, έτσι κάθε συνάρτηση και μέθοδος γνωρίζει ποιες τριάδες θα στείλει προς απεικόνιση και ποιες τριάδες θα απορρίψει, καθώς και το ότι πλέον θα δημιουργήσει τριάδες για όλα τα αξιώματα και θα απεικονίσει και τις σχέσεις με άλλες οντολογίες, όπως αυτές δηλώνονται στο C-OWL αρχείο.

6. Σενάρια χρήσης

Το εργαλείο EOKO έχει διπλή χρησιμότητα. Η πρώτη χρησιμότητα είναι η απεικόνιση και μελέτη της γενικότερης δομής μιας απλής ή κατακτημένης οντολογίας και η δεύτερη είναι η εις βάθος μελέτη της δομής μιας οντολογίας και των σχέσεων μεταξύ των κατατμημένων κομματιών της (C-OWL σχέσεων). Στη συνέχεια παρατίθενται δύο σενάρια, εκ των οποίων το κάθε ένα αναλύει τις δυνατότητες και την χρησιμότητα του εργαλείου.

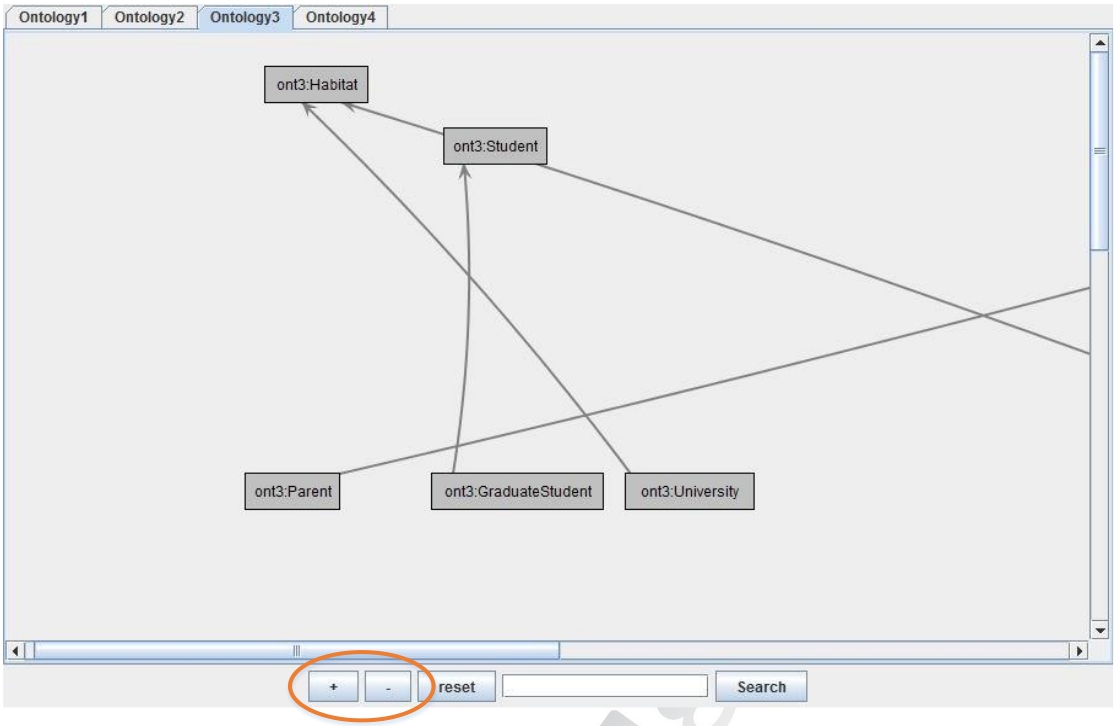
6.1.1. Σενάριο 1: Βασική απεικόνιση και μελέτη οντολογιών

Το πρώτο βήμα για να μπορέσουν να αναλυθούν οι οντολογίες είναι να εισαχθούν στο πρόγραμμα για την απεικόνισή τους. Πιο συγκεκριμένα, τοποθετούνται στο πρόγραμμα οι διευθύνσεις στις οποίες βρίσκονται οι οντολογίες μέσα στο σκληρό δίσκο και στην συνέχεια το πρόγραμμα δημιουργεί τους γράφους. Στην Εικόνα 16 δίνεται ένα παράδειγμα όπου στο εργαλείο έχουν εισαχθεί ο μέγιστος αριθμός οντολογιών (τέσσερεις οντολογίες). Κάθε οντολογία είναι τοποθετημένη σε διαφορετική καρτέλα για ευκολότερη πλοήγηση.



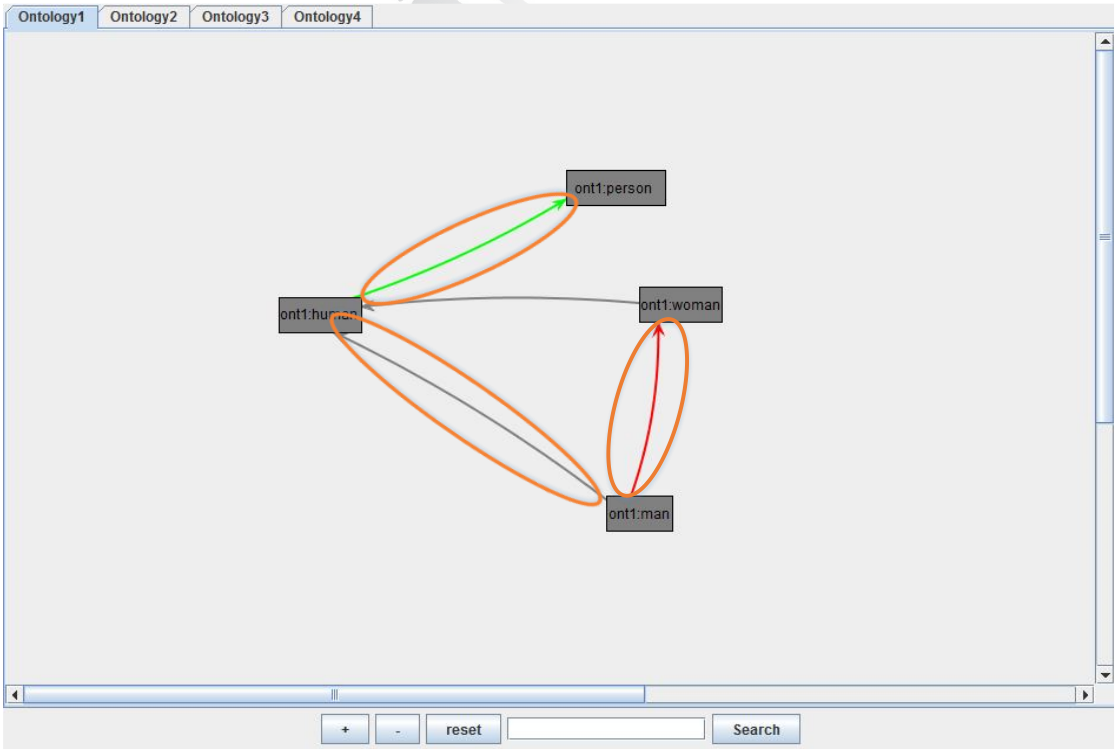
Εικόνα 16: Στιγμιότυπο γενικής διεπαφής εργαλείου EOKO.

Ο χρήστης σε αυτή τη κατάσταση του συστήματος δεν μπορεί ακόμα να μετακινήσει τους κόμβους του γράφου, μπορεί όμως να περιηγηθεί στο γράφο και να κάνει σμίκρυνση η μεγέθυνση αυτού, ώστε να δει την γενικότερη διάρθρωση των κόμβων ή τη διάρθρωση ενός συγκεκριμένου κομματιού του. Αυτό μπορεί να γίνει χρησιμοποιώντας τα πλήκτρα «+» και «-» για τη μεγέθυνση και σμίκρυνση του. Όπως φαίνεται και στην Εικόνα 17 έχει γίνει μεγέθυνση και επικέντρωση σε ένα συγκεκριμένο κομμάτι της οντολογίας.



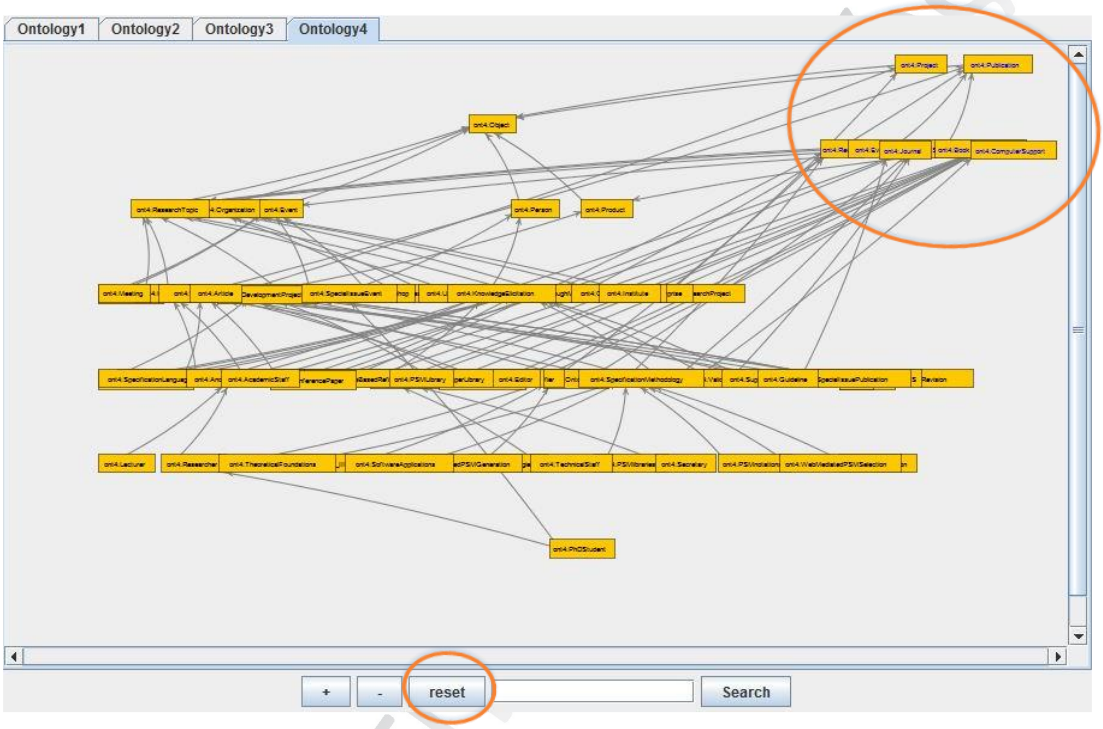
Εικόνα 17: Στιγμιότυπο λειτουργίας εστίασης στο εργαλείο EOKO

Στους γράφους που δημιουργούνται είναι εύκολο να διακριθούν οι βασικοί τύποι σχέσεων (σχέσεις υποκλάσεων, σχέσεις ίσων κλάσεων και σχέσεις ξένων κλάσεων), καθώς διακρίνονται χρωματικά. Στην Εικόνα 18 φέρεται το πώς διακρίνονται οι απλές σχέσεις υποκλάσεων με το χρώμα γκρι, οι σχέσεις ξένων κλάσεων με το χρώμα κόκκινο, καθώς και οι σχέσεις ίσων κλάσεων με το χρώμα πράσινο.



Εικόνα 18: Τρόπος εμφάνισης των βασικών σχέσεων κλάσεων από το EOKO.

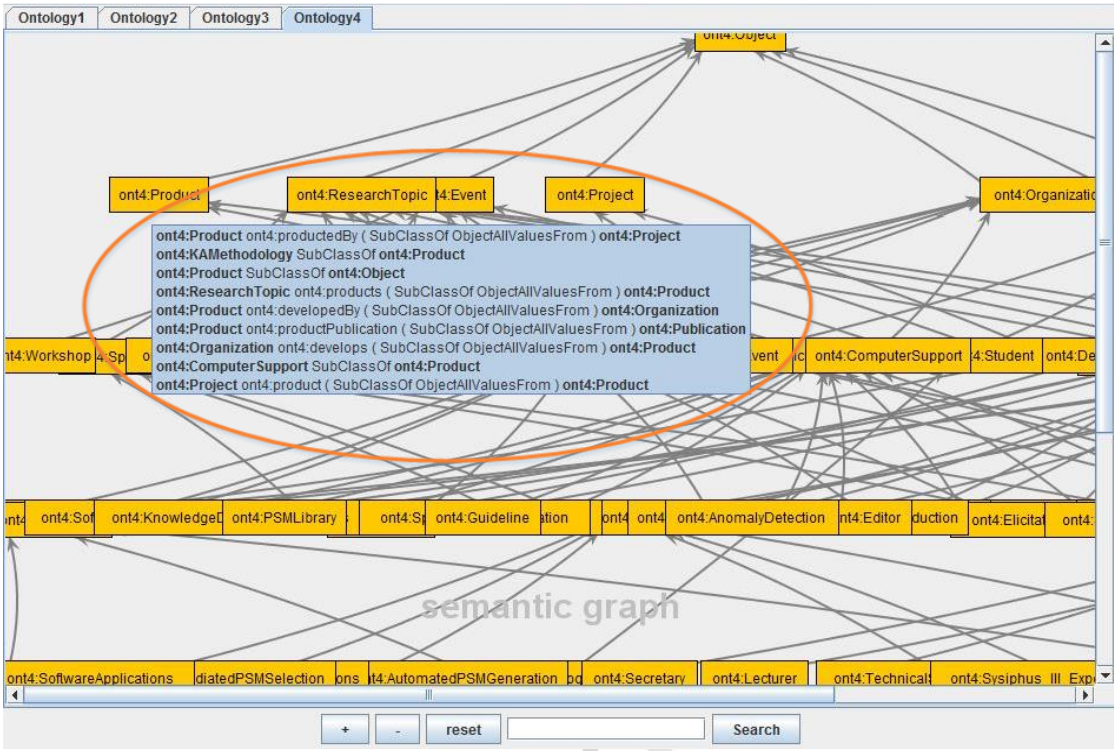
Μια οντολογία υπάρχει περίπτωση να είναι πολύ πυκνή (να περιέχει πολλούς κόμβους-κλάσεις και σχέσεις μεταξύ τους). Λόγω αυτού του φαινομένου, ο χρήστης πρέπει να έχει τη δυνατότητα να μετακινεί έναν η περισσότερους κόμβους, ώστε να γίνεται ο γράφος, καθώς και οι σχέσεις που θέλει να μελετήσει, πιο ξεκάθαροι και ευανάγνωστοι. Πατώντας το πλήκτρο «P» του πληκτρολογίου μπορεί ο χρήστης να επιλέξει έναν ή περισσότερους κόμβους και να τους μετακινήσει σε μια θέση της αρεσκείας του. Όπως φαίνεται και στην Εικόνα 19 που απεικονίζει μια σύνθετη οντολογία με πολλούς κόμβους και σχέσεις, μπορεί να μετακινηθεί μια ομάδα κόμβων ώστε να μελετηθεί ξεχωριστά.



Εικόνα 19: Στιγμιότυπο μετακίνησης πολλαπλών γράφων στο ΕΟΚΟ.

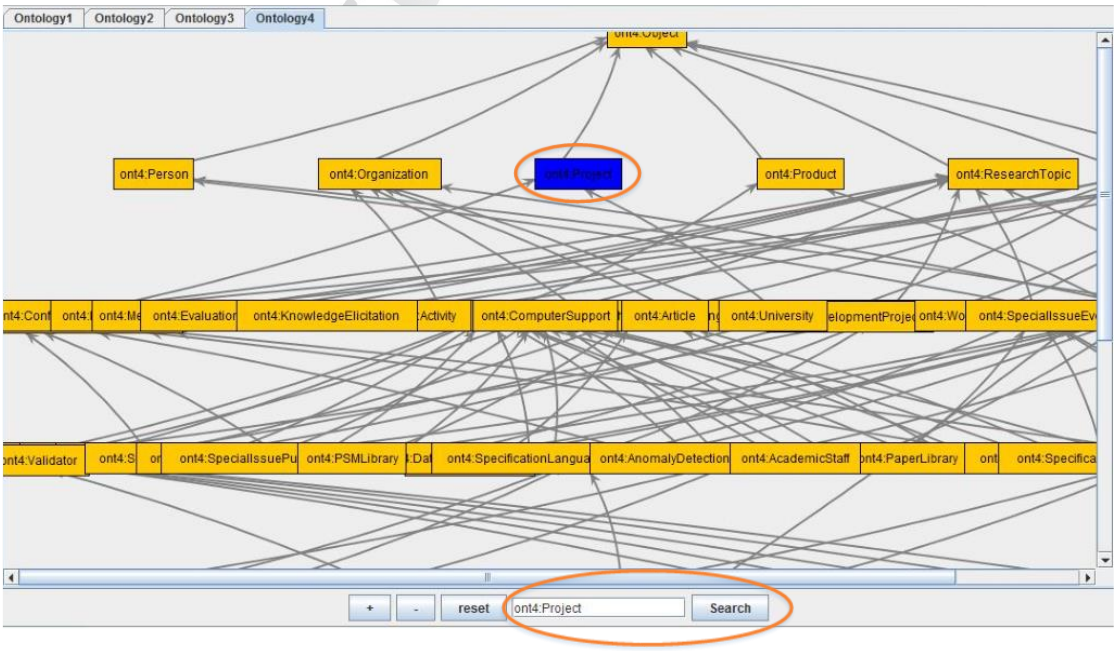
Στο χρήστη δίνεται και η επιλογή να επαναφέρει το γράφο στην αρχική του κατάσταση (Εικόνα 19). Η επιλογή αυτή δίδεται διότι ο χρήστης μπορεί να επιθυμεί να αναιρέσει αλλαγές που έχει κάνει στην διάταξη του γράφου. Σε πολλές περιπτώσεις για να μελετήσει κάποιο άλλο κομμάτι της οντολογίας, πρέπει να επανέλθει ο γράφος στην αρχική του κατάσταση. Αυτό γίνεται εύκολα και γρήγορα με το πλήκτρο «reset» που βρίσκεται στο κάτω μέρος του παραθύρου.

Οι κόμβοι και οι συσχετίσεις τους, δεν είναι η μόνη πληροφορία που παρατίθενται προς τον χρήστη. Όπως αναφέρθηκε και στην προηγούμενη παράγραφο οι οντολογίες μπορεί να είναι πολύ πυκνές, λόγω αυτού του γεγονότος δίδεται στο χρήστη ακόμα μια λειτουργική υποβοήθηση για τη καλύτερη μελέτη της οντολογίας που απεικονίζεται. Όπως δείχνει η Εικόνα 20 , αν ο χρήστης τοποθετήσει τον κέρσορά του πάνω από ένα κόμβο, εμφανίζεται ένα παράθυρο που δείχνει όλες της σχέσεις που αφορούν αυτό τον κόμβο μέσα στην οντολογία. Με αυτό τον τρόπο ο χρήστης μπορεί να αντλήσει πληροφορία που χρειάζεται χωρίς να επέμβει στη μορφή του γράφου.



Εικόνα 20: Στιγμιότυπο τοποθέτησης κέρσορα πάνω από κόμβο του γράφου .

Σε μια μεγάλη οντολογία είναι πιθανό να μην είναι πλήρως διακριτός ή να μπορεί εύκολα να εντοπιστεί κάποιος κόμβος. Για το λόγο αυτό υπάρχει η δυνατότητα αναζήτησης κόμβων. Ο χρήστης γράφει με συντομογραφία σε ποια από τις εικονιζόμενες οντολογίες θέλει να ψάξει (πχ «ont1:» για να ψάξει το εργαλείο στη πρώτη οντολογία, «ont2:» για να ψάξει το εργαλείο στη δεύτερη οντολογία κ.ο.κ.) και στη συνέχεια το όνομα του κόμβου που ψάχνει. Αν το εργαλείο βρει αντιστοιχία, χρωματίζει το κόμβο με σκούρο μπλε χρώμα ώστε να εντοπιστεί εύκολα από το χρήστη (Εικόνα 21).

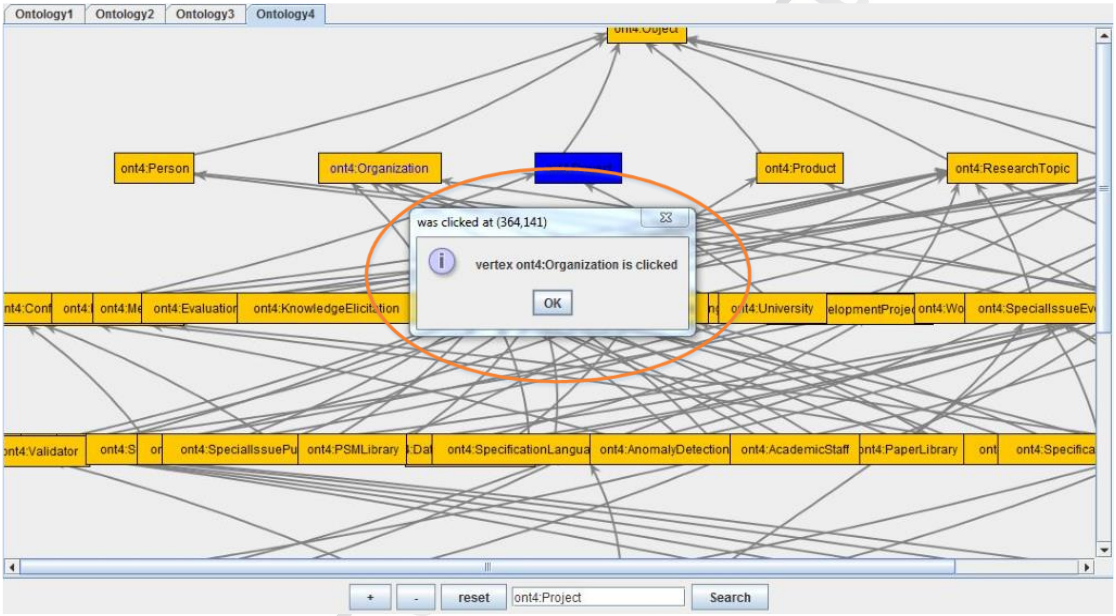


Εικόνα 21: Στιγμιότυπο λειτουργίας αναζήτησης του ΕΟΚΟ

6.1.2. Σενάριο 2: Εις βάθος μελέτη οντολογίας και συσχετίσεων οντολογιών

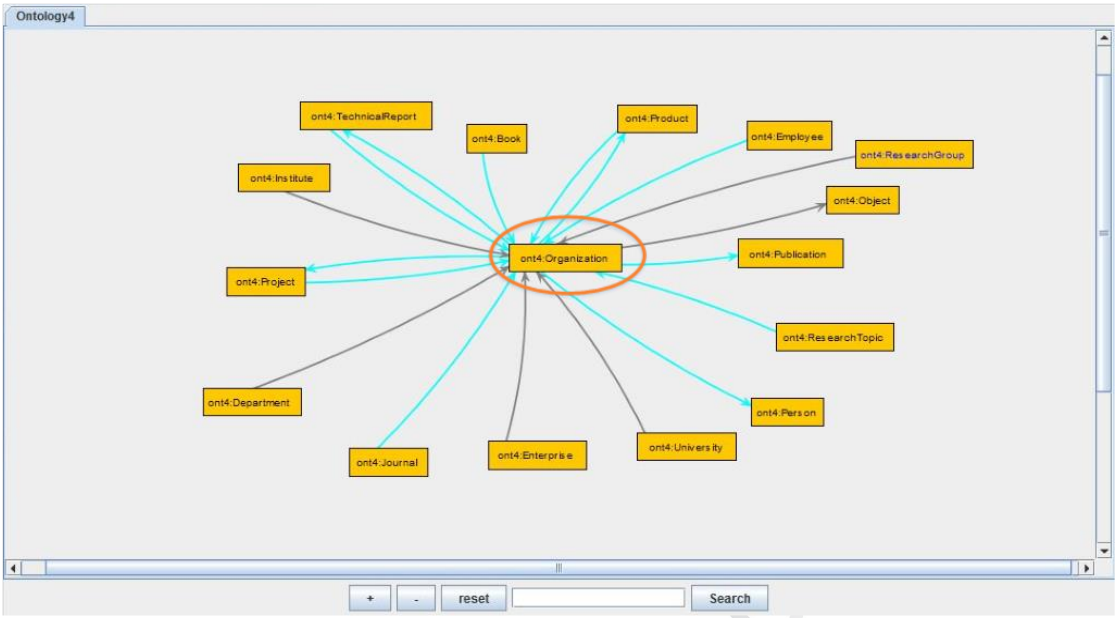
Όπως αναφέρθηκε και στην αρχή του κεφαλαίου το EOKO μπορεί να χρησιμοποιηθεί για την εις βάθος ανάλυση οντολογιών. Στο πρώτο σενάριο αναλύθηκαν οι βασικές λειτουργείες του EOKO, ενώ σε αυτό το σενάριο αναλύεται ο τρόπος που μπορούν να μελετηθούν εις βάθος μια ή περισσότερες οντολογίες, χρησιμοποιώντας το EOKO.

Έχοντας δει και μελετήσει τον αρχικό γράφο της οντολογίας ο χρήστης έχει τη δυνατότητα να επιλέξει ένα συγκεκριμένο κόμβο, ώστε να δημιουργηθεί ένας επιπρόσθετος γράφος που παρουσιάζει όλη τη πληροφορία (σχέσεις) που αφορά αυτόν συγκεκριμένα το κόμβο. Όπως φαίνεται και στην Εικόνα 22, ο χρήστης μόλις επιλέξει κάποιο κόμβο (στη συγκεκριμένη περίπτωση το κόμβο Organization), το εργαλείο εμφανίζει ένα μήνυμα επιβεβαίωσης. Μόλις ο χρήστης πατήσει το κουμπί «OK» δημιουργείται ο καινούριος γράφος.



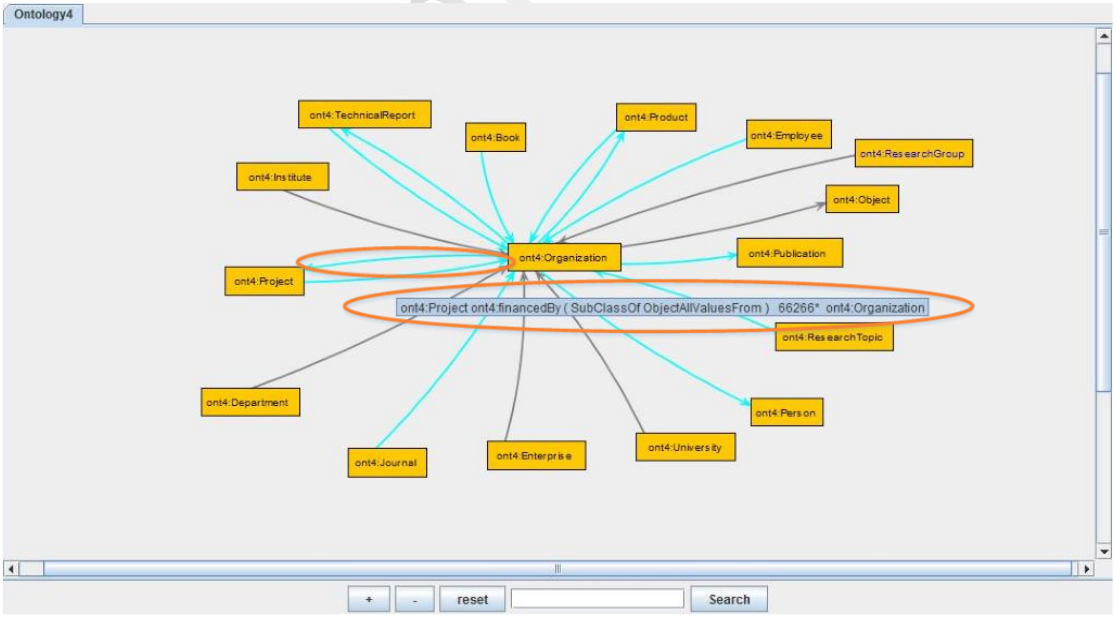
Εικόνα 22: Στιγμιότυπο επιλογής κόμβου στο γράφο.

Στον καινούριο γράφο το εργαλείο τοποθετεί τον επιλεγμένο κόμβο στο κέντρο του γράφου και γύρω από αυτόν, τους κόμβους με τους οποίους συνδέεται (Εικόνα 23).



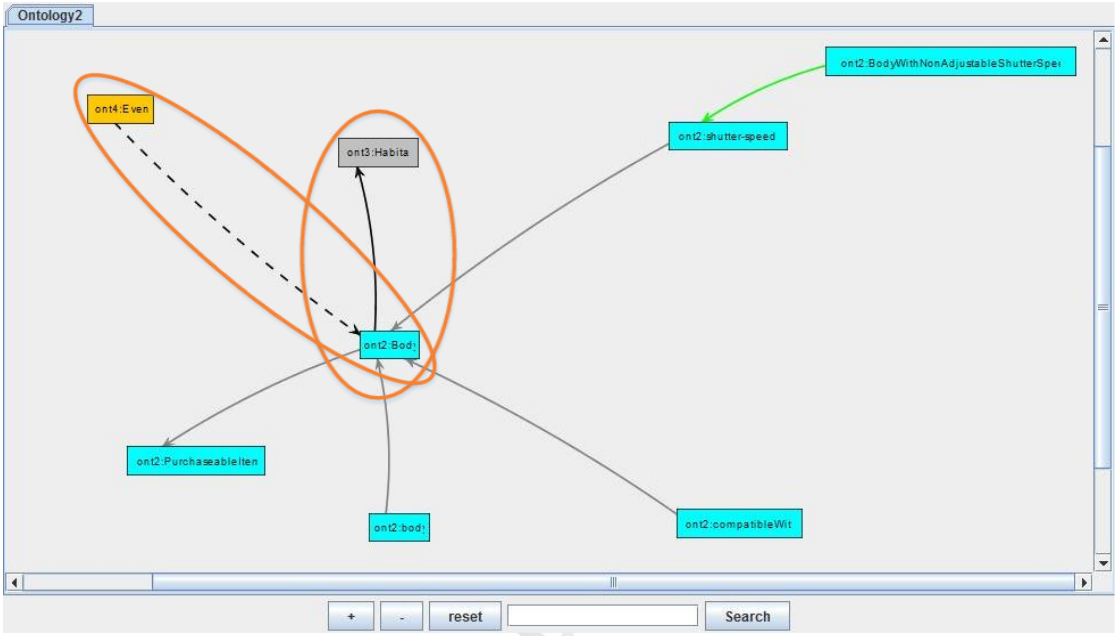
Εικόνα 23: Στιγμιότυπο αναλυτικού γράφου επιλεγμένου κόμβου στο EOKO.

Στον καινούριο γράφο δεν εμφανίζονται μόνο οι απλές σχέσεις όπως στον αρχικό, εμφανίζονται όλες οι σχέσεις που αφορούν το κόμβο που επιλέχθηκε. Για να ξεχωρίζουν οι ακμές που αναπαριστούν ιδιότητες ή άλλου τύπου σχέσεις, το πρόγραμμα τις χρωματίζει γαλάζιες. Αν ο χρήστης τοποθετήσει τον κέρσορα πάνω από αυτές τις ακμές εμφανίζεται ένα παράθυρό που περιγράφει πλήρως την σχέση που αναπαριστά αυτή η ακμή. Με αυτό τον τρόπο ο χρήστης μπορεί να μελετήσει όλες τις σχέσεις που υπάρχουν στο γράφο χωρίς να χρειάζεται η δημιουργία ενός πολύ περίπλοκου και επιβαρυμένου με πλεονάζουσα πληροφορία γράφου (Εικόνα 24).



Εικόνα 24: Στιγμιότυπο τοποθέτησης κέρσορα πάνω από μια ακμή του γράφου.

Μια σημαντική λειτουργία του EOKO είναι η αναπαράσταση των C-OWL σχέσεων. Η αναπαράσταση αυτή λαμβάνει χώρα μέσα στο γράφο που παρουσιάζει όλες τις σχέσεις ενός επιλεγμένου κόμβου. Εάν ο χρήστης έχει εισάγει στο εργαλείο ένα C-OWL αρχείο, το πρόγραμμα το προσπελαύνει για να εντοπίσει σχέσεις μέσα σε άλλες οντολογίες που αφορούν αυτό το κόμβο. Με αυτό τον τρόπο στο γράφο δεν θα εμφανιστούν μόνο οι σχέσεις του κόμβου μέσα στην οντολογία του, αλλά και οι σχέσεις που έχει με άλλες οντολογίες.



Εικόνα 25: Στιγμιότυπο απεικόνισης C-OWL σχέσεων μέσα στον αναλυτικό γράφο του EOKO.

Όπως φαίνεται και στην Εικόνα 25, οι δυο κόμβοι Habitat (γκρι) και Event (πορτοκαλί) έχουν διαφορετικό χρώμα. Αυτό συμβαίνει διότι αναπαριστούν έννοιες διαφορετικών οντολογιών. Οι κόμβοι αυτοί παίρνουν το χρώμα της οντολογίας που ανήκουν για να είναι εύκολα διακριτοί. Οι ακμές των C-OWL σχέσεων έχουν μαύρο χρώμα για να διαφέρουν από τις σχέσης OWL. Αν η ακμή απεικονίζει σχέση υποκλάσης τότε παίρνει το χρώμα μαύρο (π.χ. Εικόνα 25 κλάση:Habitat). Για σχέσεις ιδιοτήτων και περιορισμών η ακμή έχει χρώμα μαύρο και είναι διακεκομμένη(π.χ. Εικόνα 25 κλάση:Event).

7. Συμπεράσματα

Σε αυτό το κεφάλαιο παρατίθενται τα συμπεράσματα που αφορούν στην υλοποίηση και λειτουργία του ΕΟΚΟ. Αρχικά συγκρίνεται το ΕΟΚΟ με τα υπάρχοντα εργαλεία απεικόνισης οντολογιών. Στη συνέχεια ακολουθούν τα συμπεράσματα για τις προγραμματιστικές επιλογές που έγιναν και τελειώνοντας αναφέρονται πιθανές εξελίξεις του εργαλείου.

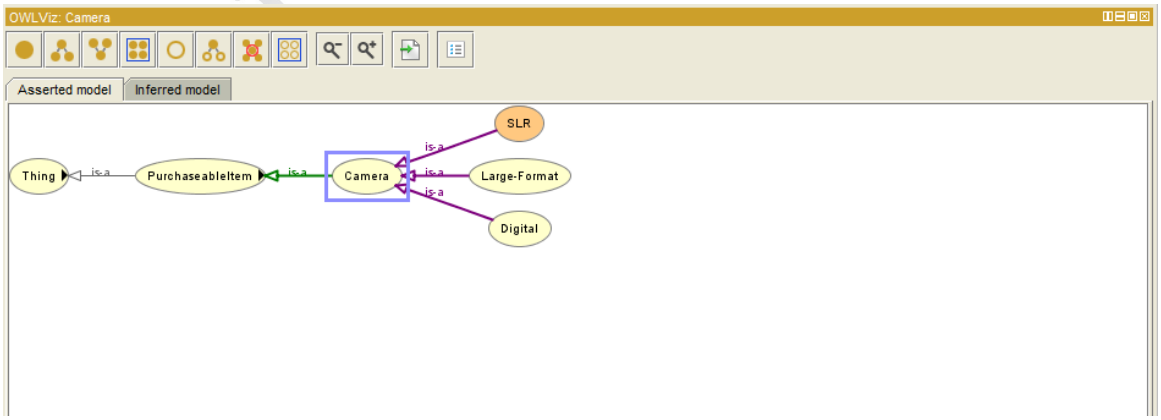
7.1. Τεχνολογική αιχμή

Στο κεφάλαιο αυτό επιχειρείται μια τυπική ανάλυση των επικρατέστερων εργαλείων απεικόνισης και ανάλυσης οντολογιών, που έχουν δημιουργηθεί ως τώρα. Η ανάλυση αυτή βοηθάει στη κατανόηση του τρόπου αναπαράστασης της πληροφορίας από αυτά τα εργαλεία, καθώς και στην συνεισφορά και τη διαφοροποίηση του Εργαλείου Οπτικοποίησης Κατατμημένων Οντολογιών από αυτά.

7.1.1. Εργαλείο Protégé

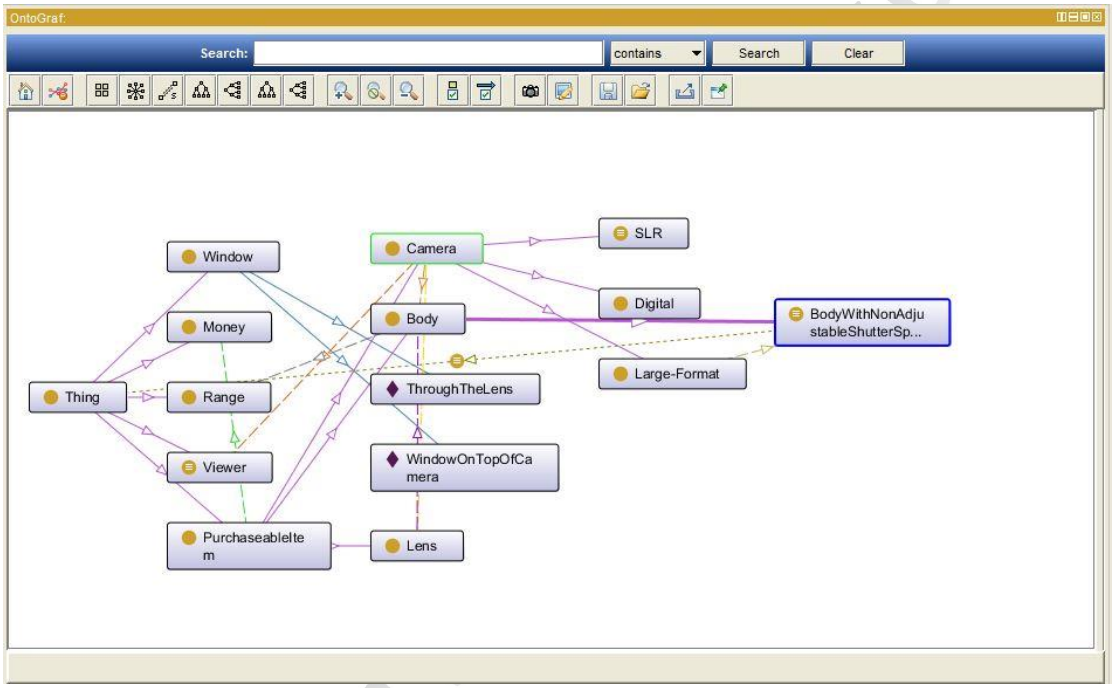
Το Protégé είναι ένα δωρεάν εργαλείο ανοιχτού κώδικα που δημιουργήθηκε από το πανεπιστήμιο Stanford για την δημιουργία και την επεξεργασία οντολογιών. Η κύρια λειτουργία του Protégé αφορά στη δημιουργία και επεξεργασία μιας οντολογίας˙ παρόλα αυτά, είναι ενισχυμένο με δύο επιπλέον εργαλεία για την απεικόνιση οντολογιών. Τα δύο αυτά εργαλεία είναι το OWLViz και το OntoGraph.

Κάθε εργαλείο δημιουργεί διαφορετική αναπαράσταση των οντολογιών και κάθε αναπαράσταση στοχεύει στο να αναλύσει διαφορετικά το περιεχόμενο της οντολογίας. Το OWLViz είναι βασισμένο στο GraphViz και δημιουργεί έναν αφαιρετικό γράφο για κάθε στοιχείο της οντολογίας (ώστε να μελετηθεί ξεχωριστά) και όχι το γράφο ολόκληρης της οντολογίας (Εικόνα 26). Όπως φαίνεται και στην Εικόνα 26 το εργαλείο παρουσιάζει τις σχέσεις που αφορούν μια συγκεκριμένη κλάση και πάνω στις ακμές φαίνεται ο τύπος της κάθε σχέσης. Με αυτό τον τρόπο το OWLViz προσπαθεί να πετύχει λεπτομερειακή ανάλυση των οντολογιών(απεικόνιση μιας κλάσης και των σχέσεών της σε έναν ξεχωριστό γράφο χωρίς την απεικόνιση όλης της οντολογίας), χωρίς να δημιουργεί γράφους με ιδιαίτερη λειτουργικότητα και διαδραστικότητα, αναπαριστώντας παρ' όλα αυτά όλη την σημαντική πληροφορία(όλες τις κλάσεις και σχέσεις που υπάρχουν μεταξύ τους) για μια ολοκληρωμένη ανάλυση [6].



Εικόνα 26: Εργαλείο απεικόνισης οντολογιών OWLViz.

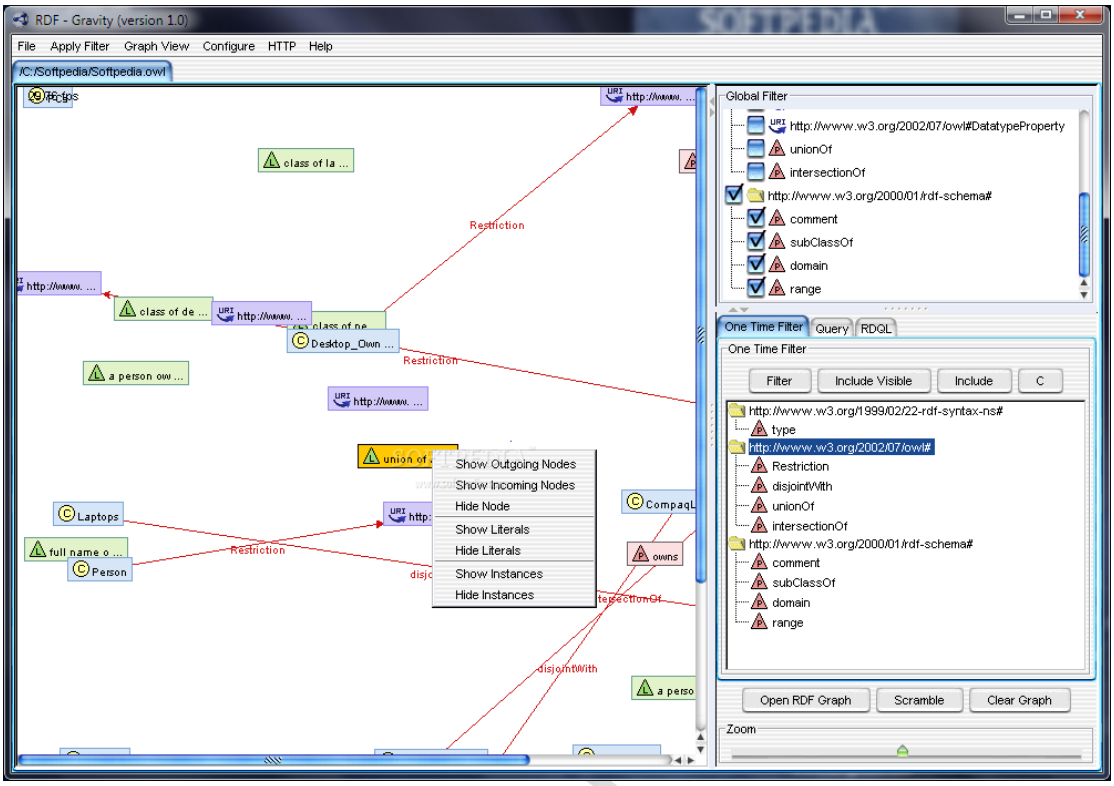
Το εργαλείο OntoGraph δημιουργεί ένα πιο διαδραστικό και γενικό γράφο όλης της οντολογίας (Εικόνα 27) σε σχέση με το OWLViz. Όλες οι κλάσεις και οι υποκλάσεις, καθώς και οι σχέσεις τους απεικονίζονται σε ένα γενικότερο γράφο. Υπάρχει η δυνατότητα διάδρασης με το γράφο, πιο συγκεκριμένα υποστηρίζει την ανάπτυξη των γονικών κλάσεων ώστε να εμφανιστούν οι υποκλάσεις τους, μετακίνηση των κόμβων, δυνατότητα αναζήτησης κόμβων και δυνατότητα σύμπτυξης κλάσεων και εμφάνιση μόνο των γονικών όπως φαίνεται και στην Εικόνα 27. Ο γράφος που δημιουργεί το OntoGraph μπορεί να είναι διαδραστικός (μετακίνηση κόμβων, επέκταση περιεχομένου υπερκλάσεων) και να απεικονίζει όλη τη πληροφορία, αλλά δεν υποστηρίζει την απεικόνιση μεμονωμένων κλάσεων και σχέσεών τους όπως δημιουργεί το OWLViz[6].



Εικόνα 27: Εργαλείο απεικόνισης οντολογιών OntoGraph.

7.1.2. Εργαλείο RDF-Gravity

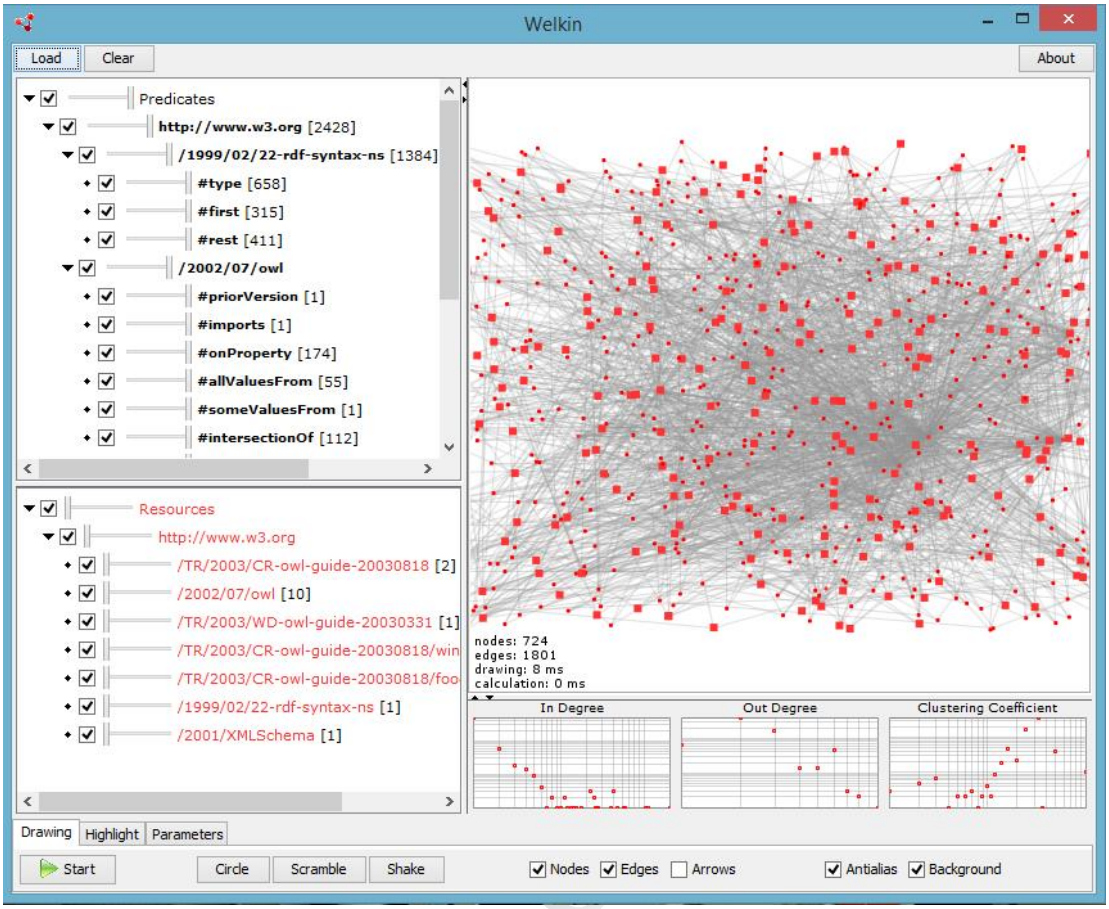
Το εργαλείο RDF-Gravity αποτελεί ένα από τα πλέον διαδεδομένα εργαλεία οπτικοποίησης οντολογιών. Είναι αναπτυγμένο σε γλώσσα Java και χρησιμοποιεί τη βιβλιοθήκη Jung για την δημιουργία των γράφων. Υποστηρίζει την αναπαράσταση RDF και OWL αρχείων. Το εργαλείο παρέχει μια απλή, αλλά πολύ ισχυρή απεικόνιση των οντολογιών. Το μεγάλο πλεονέκτημα του εργαλείου RDF-Gravity είναι η δυνατότητα φιλτραρίσματος και απεικόνισης συγκεκριμένης πληροφορίας στο γράφο. Η λειτουργικότητα και η διάδραση στο γράφο είναι επαρκείς για μια εκτεταμένη ανάλυση μιας οντολογίας. Ένα ακόμα σημαντικό χαρακτηριστικό του RDF-Gravity είναι η πλούσια σήμανση των στοιχείων στο γράφο. Λόγω της πληθώρας πληροφοριών που απεικονίζει το RDF-Gravity, υπάρχει πολύ πλούσια σήμανση (άλλο χρώμα και σχήμα για κάθε τύπο σχέσης (ακμής) και κάθε τύπο κλάσης (κόμβου)) των στοιχείων που απεικονίζονται στο γράφο, ώστε να μπορεί ο χρήστης εύκολα να μελετήσει τις σχέσεις μέσα στην οντολογία, καθώς επίσης, να κάνει τα διάφορα στοιχεία που απεικονίζονται ευδιάκριτα. (Εικόνα 28). Το μειονέκτημα του RDF-Gravity είναι ότι παρά την πλούσια σήμανση, μπορεί να δημιουργήσει πολύ περίπλοκους γράφους με πολλούς κόμβους και ακμές[7].



Εικόνα 28: Εργαλείο απεικόνισης οντολογιών RDF-Gravity.

7.1.3. Εργαλείο Welkin

Το Welkin είναι ένα εργαλείο που αναπτύχθηκε από το MIT (Massachusetts Institute of Technology). Όπως και το RDF-Gravity, είναι προγραμματισμένο σε Java. Το Welkin είναι ένα διαδραστικό εργαλείο που δημιουργεί έναν γενικό αφαιρετικό γράφο(παρουσιάζει μόνο κλάσεις και βασικές σχέσεις αυτών όπως σχέσεις υποκλάσεων), χωρίς να παραθέτει πολλές πληροφορίες για τα αντικείμενα που απεικονίζει. Απεικονίζονται όλα τα αντικείμενα και οι σχέσεις τους σε έναν ενιαίο γράφο όπως φαίνεται και στην Εικόνα 29. Δεν υπάρχει ιδιαίτερη διάδραση με το γράφο και δεν τροποποιείται ιδιαίτερα πέρα από τη μετακίνηση των κόμβων και την αλλαγή του βάθους της ανάλυσης του. Παραθέτει σημαντική πληροφορία για τις πηγές των πληροφοριών που εμφανίζονται στο γράφο (URI's, domains), όχι όμως πάνω στους κόμβους, αλλά σαν γενικά στατιστικά στοιχεία σε έναν πίνακα έξω από το γράφο (Εικόνα 29 αριστερό μέρος)[8].



Εικόνα 29: Εργαλείο απεικόνισης οντολογιών Welkin

7.2. Διαφοροποίηση του EOKO από τα άλλα εργαλεία απεικόνισης οντολογιών

Η κύρια διαφορά του EOKO από τα άλλα εργαλεία απεικόνισης οντολογιών είναι η επεξεργασία και απεικόνιση των C-OWL σχέσεων για την απεικόνιση κατατμημένων οντολογιών. Το EOKO δημιουργήθηκε για αυτόν κύρια το σκοπό και αποτελεί τη κύρια λειτουργία του. Όλα τα εργαλεία απεικόνισης οντολογιών επεξεργάζονται μόνο RDF-XML και OWL αρχεία και έτσι δεν είναι δυνατή η απεικόνιση σχέσεων μεταξύ οντολογιών. Μια ακόμη σημαντική διαφορά του EOKO από τα αλλά εργαλεία είναι η απεικόνιση πολλών οντολογιών ταυτόχρονα. Όλα τα εργαλεία υποστηρίζουν την εμφάνιση μια οντολογίας κάθε φορά. Η απεικόνιση πολλών οντολογιών στοχεύει και αυτή στην καλύτερη μελέτη των κατατμημένων οντολογιών, διότι όταν υπάρχουν σχέσεις μεταξύ οντολογιών μέσα στο γράφο είναι πολύ σημαντικό να μπορεί ο χρήστης να βλέπει τα κομμάτια των κατατμημένων οντολογιών αυτοτελή, ώστε να κατανοεί και να μπορεί να αναλύσει καλύτερα από πού προκύπτουν οι διάφορες συσχετίσεις. Το EOKO υποστηρίζει τη μεμονωμένη μελέτη στοιχείων της οντολογίας. Το χαρακτηριστικό αυτό το υποστηρίζει και το OWLViz, χωρίς όμως να υπάρχει αλληλεπίδραση με το γράφο ή να υποστηρίζει την απεικόνιση γενικού γράφου. Το EOKO υποστηρίζει μεγάλου βαθμού αλληλεπίδραση με τους γράφους που δημιουργεί, όπως και τα περισσότερα εργαλεία. Η αλληλεπίδραση με το γράφο έχει βασιστεί σε μεγάλο βαθμό στο τρόπο αλληλεπίδρασης που παρέχει το εργαλείο του Protégé OntoGraph, που δίνει τη δυνατότητα μετακίνησης κόμβων καθώς και παρουσίασης πληροφορίας με το πέρασμα του κέρσορα πάνω από τους κόμβους και

τις ακμές. Το ΕΟΚΟ δίνει επιπλέον τη δυνατότητα αναζήτησης κόμβων στα πλαίσια της αλληλεπίδρασης με το γράφο. Το μόνο από τα υπάρχοντα εργαλεία που επιτρέπει αναζήτηση κόμβων είναι το OntoGraph. Στον παρακάτω πίνακα (Εικόνα 30) παρουσιάζεται η σύγκριση των κύριων χαρακτηριστικών των επικρατέστερων εργαλείων απεικόνισης οντολογιών, καθώς και του ΕΟΚΟ. Τα χαρακτηριστικά στα οποία εξετάστηκαν όλα τα εργαλεία έχουν αντληθεί από τους στόχους και τις απαιτήσεις που αναλύθηκαν στο Κεφάλαιο 3 για τη δημιουργία του εργαλείου ΕΟΚΟ.

Εικόνα 30: Συγκριτικός πίνακας λειτουργιών εργαλείων απεικόνισης Οντολογιών.

7.3. Συμπεράσματα προγραμματιστικών επιλογών και υλοποίησης του ΕΟΚΟ

Τα συμπεράσματα των προγραμματιστικών επιλογών αφορούν τις βιβλιοθήκες και τις διεπαφές προγραμματιστικών εφαρμογών (API) που χρησιμοποιήθηκαν. Οι κύριες διεπαφές που χρησιμοποιήθηκαν, όπως έχουμε προαναφέρει, είναι το OWL-API, η Jung και η Dom4j.

Το OWL-API αποτελεί μια πολύ ισχυρή διεπαφή με πολλές λειτουργίες και συναρτήσεις, που βοηθούν στην αποδοτική και σωστή ανάλυση οντολογιών. Ωστόσο, είναι αρκετά περίπλοκη, με κακό έγγραφο τεκμηρίωσης. Η εύκολη εξαγωγή των λογικών αξιωμάτων αποτελεί το σημαντικότερο χαρακτηριστικό της, αλλά είναι πολύ δύσκολη η ανάλυσή των αξιωμάτων χρησιμοποιώντας το OWL-API. Η πολυπλοκότητα των εννοιών ενός αξιώματος ποικίλει. Για αυτό το λόγο, χρειάζεται μια αναδρομική μέθοδος ανάλυσης σύνθετων εννοιών μέχρι να γίνουν απλές, ο χειρισμός όμως από το OWL-API δεν καθιστά εφικτή τη δημιουργία μιας τέτοιας αναδρομικής μεθόδου. Η λύση σε αυτό το πρόβλημα ήταν να μετατραπούν τα αξιώματα σε συμβολοακολουθίες και μετέπειτα να γίνει ανάλυση αυτών των συμβολοακολουθιών με βάση το φορμαλισμό γραφής τους. Αυτός ο τρόπος ανάλυσης των αξιωμάτων είναι λιγότερο αποδοτικός σε ό,τι αφορά την υπολογιστική πολυπλοκότητα, διότι τα αξιώματα αναλύονται(χρησιμοποιώντας την βιβλιοθήκη

OWLAPI) με βάση τα στοιχεία(κλάσεις, σχέσεις) που περιέχουν σε αντίθεση με τις συμβολοακολουθίες που αναλύεται κάθε σύμβολο που υπάρχει μέσα σε αυτές. Με αυτή τη μέθοδο παρόλα αυτά διασφαλίζεται η σωστή ανάλυση των αξιωμάτων.

Η διεπαφή Jung αποτελεί εύκολο και γρήγορο μέσο για τη δημιουργία γράφων, χρησιμοποιείται ευρέως και είναι η διεπαφή που χρησιμοποιεί το RDF-Gravity, καθώς επίσης υποστηρίζεται από ένα πλήρες έγγραφο τεκμηρίωσης. Η Jung δίνει την δυνατότητα προσθήκης πολλών λειτουργικών χαρακτηριστικών, που χρησιμοποιήθηκαν και στο EOKO (π.χ. διάδραση με γράφο, κουμπιά λειτουργιών, μεγέθυνσης, σμίκρυνσης). Ωστόσο η Jung έχει ένα μειονέκτημα. Όσο αυξάνεται η λειτουργικότητα και η διαδραστικότητα στο γράφο, δημιουργούνται προβλήματα στην απεικόνισή του. Πιο συγκεκριμένα, όταν πρόκειται να απεικονίσει το Jung μια πυκνή και μεγάλη οντολογία, δημιουργούνται προβλήματα στην διάταξη του γράφου καθώς αυτή χάνεται και οι κόμβοι τείνουν να συγκλίνουν προς το πάνω μέρος του παραθύρου, μετατρέποντας το γράφο σε ένα συνονθύλευμα ενωμένων κόμβων στην άκρη του παραθύρου. Το φαινόμενο αυτό συμβαίνει διότι ο όγκος των δεδομένων είναι πολύ μεγάλος και αδυνατεί η προσωρινή μνήμη που δεσμεύεται για τη λειτουργία του προγράμματος να αποθηκεύσει όλη αυτή τη πληροφορία. Δοθέντος αυτού του προβλήματος, εισήχθη ο περιορισμός να μπορούν να απεικονιστούν το πολύ τέσσερεις οντολογίες ταυτόχρονα, ώστε να αντιμετωπιστούν τα προβλήματα που προαναφέρθηκαν.

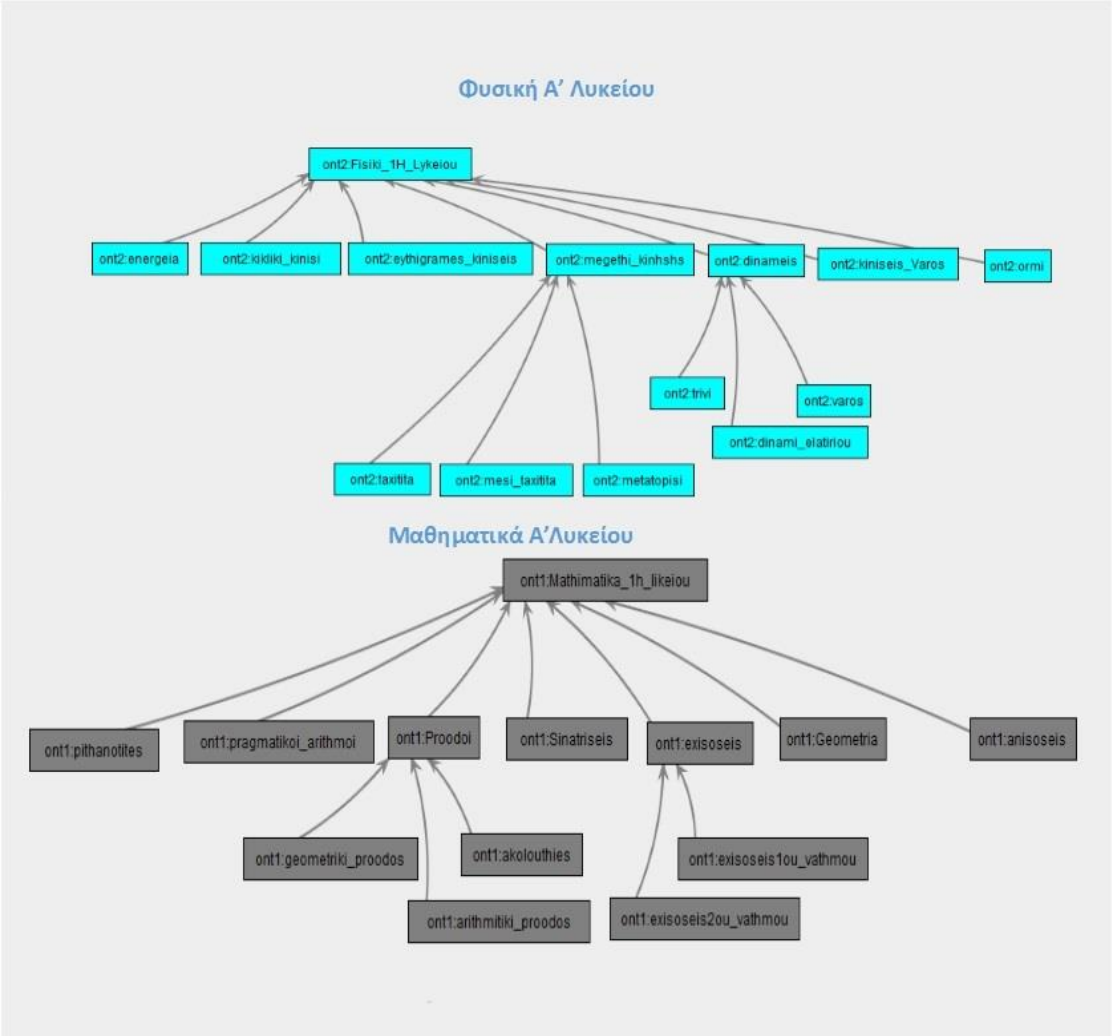
Η βιβλιοθήκη Dom4j αποτελεί την πιο διαδεδομένη βιβλιοθήκη για την ανάλυση XML αρχείων. Η Dom4j έχει απλό συντακτικό εντολών και μπορούν να εξαχθούν εύκολα και αποδοτικά αντικείμενα από XML αρχεία. Η Dom4j δεν συνιστάται για περιπλοκή και μεγάλου βάθους ανάλυση XML αρχείων, αντιθέτως για τους σκοπούς της ανάλυσης C-OWL αρχείων είναι ιδανική και καλύπτει όλες τις ανάγκες ανάλυσης των αρχείων.

7.4. Εκπαιδευτική χρήση του EOKO

Το εργαλείο EOKO δεν δημιουργήθηκε για εκπαιδευτικούς σκοπούς, ωστόσο η δημιουργία του μπορεί να βοηθήσει στην ανάπτυξη συνδέσεων μεταξύ οντολογιών που αφορούν εκπαιδευτικούς σκοπούς.

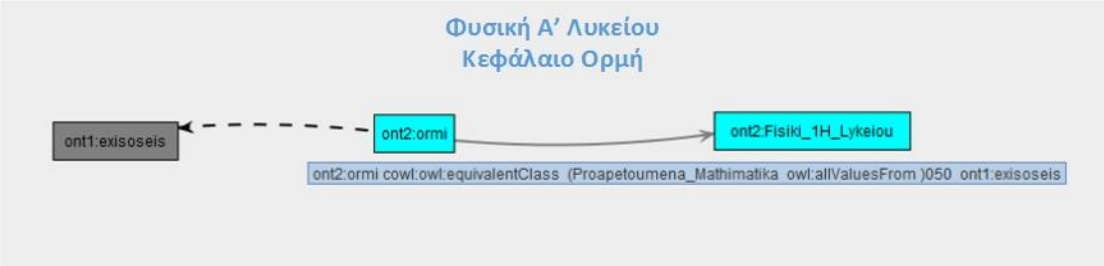
Οι οντολογίες αποτελούν μια ευρέως διαδεδομένη και ταχέως αναπτυσσόμενη τεχνολογία μεταδεδομένων. Αυτή η τεχνολογία μπορεί να χρησιμοποιηθεί για να περιγράψει αλλά και να δώσει σημασιολογία σε πολλούς εκπαιδευτικούς τομείς. Στην συνέχεια θα παρουσιαστεί ένα σενάριο που δείχνει μια δυνητική χρήση των οντολογιών στην εκπαιδευτική διαδικασία και το πώς οι γράφοι απεικόνισης κατατμημένων οντολογιών μπορούν να την ενισχύσουν.

Κάθε μάθημα μιας σχολικής τάξης έχει μια βασική δομή που μπορεί να περιγραφεί σε μια οντολογία. Έστω ότι περιγράφονται για τις ανάγκες του παραδείγματος τα μαθήματα φυσική Α’ τάξης Λυκείου και μαθηματικά Α’ Λυκείου. Οι οντολογίες που περιγράφουν την δομή αυτών των μαθημάτων μπορούν να αναπαρασταθούν με το EOKO όπως φαίνεται και στην Εικόνα 31.



Εικόνα 31:Απεικονίσεις οντολογιών μαθημάτων Α' Λυκείου.

Όπως φαίνεται και στην Εικόνα 31 απεικονίζεται ξεκάθαρα η δομή των δύο μαθημάτων. Σημαντικό είναι να ξεκαθαριστεί ότι δεν παρουσιάζεται η πλήρης δομή των μαθημάτων (όλα τα κεφάλαια, υποκεφάλαια, παράγραφοι, ασκήσεις). Ωστόσο με την απεικόνιση σε γράφο γίνεται πολύ πιο κατανοητή η δομή ενός μαθήματος και το πως διαρθρώνεται η ύλη του. Πηγαίνοντας την ανάλυση και την περιγραφή των μαθημάτων ένα βήμα παραπέρα, μπορεί το εργαλείο να απεικονίσει και τις σχέσεις των μαθημάτων. Πιο συγκεκριμένα ένας μαθητής για να μπορέσει να κατανοήσει και να λύσει ασκήσεις που αφορούν το κεφάλαιο της ορμής (φυσική Α' Λυκείου) είναι απαραίτητο να έχει μεγάλη εξοικείωση στην λύση εξισώσεων(μαθηματικά Α' Λυκείου). Συνεπώς υπάρχει σύνδεση μεταξύ των δυο οντολογιών. Το κεφάλαιο της ορμής έχει προαπαιτούμενο το κεφάλαιο των μαθηματικών εξισώσεις. Χρησιμοποιώντας την C-OWL μπορεί να περιγραφεί αυτή η σχέση και χρησιμοποιώντας το ΕΟΚΟ μπορεί να απεικονιστεί(Εικόνα 32).



Εικόνα 32: Σχέση μεταξύ κεφαλαίου ορμής και κεφαλαίου εξισώσεων.

Όπως φαίνεται στην Εικόνα 32 το ΕΟΚΟ απεικονίζει την σχέση μεταξύ των στοιχείων «ορμή» και «εξισώσεις» των δυο οντολογιών μέσω της ιδιότητας «προαπαιτούμενα μαθηματικά». Ο συνδυασμός της περιγραφής των μαθημάτων χρησιμοποιώντας οντολογίες, καθώς και η απεικόνισή τους, παράλληλα με την δημιουργία συνδέσεων μεταξύ των μαθημάτων, μπορεί να γίνει ένα πολύ χρήσιμο εργαλείο μελλοντικά για την οργάνωση και κατανόηση της δομής και των σχέσεων των μαθημάτων· τόσο για τους διδάσκοντες όσο και για τους μαθητές.

7.5. Εξέλιξη εργαλείου

Το εργαλείο ΕΟΚΟ έχει αρκετές δυνατότητες για εξέλιξη και προσθήκη νέων χαρακτηριστικών. Μια σημαντική επέκταση του ΕΟΚΟ είναι να υποστηρίζει λογικούς συλλογισμούς (reasoning) για τις οντολογίες που απεικονίζει, χρησιμοποιώντας «reasoner» όπως ο HermiT και ο FaCT++ για τις OWL οντολογίες[12] και το πλαίσιο απεικόνισης E-SHIQ για τις C-OWL σχέσεις [4] . Ακόμα το εργαλείο μπορεί να εξελιχθεί με το να δημιουργεί επιπλέον γράφους που θα δημιουργούνται με βάση απαντήσεων SPARQL ερωτημάτων, που θα έχει τη δυνατότητα να εκτελεί ο χρήστης. Μια ακόμα σημαντική εξέλιξη του ΕΟΚΟ είναι η επεξεργασία της οντολογίας μέσω των γράφων που δημιουργεί. Πιο συγκεκριμένα να δίδεται στο χρήστη η δυνατότητα να εισάγει καινούριους κόμβους ή ακμές στο γράφο και αυτές να μεταφράζονται στη συνέχεια σε γλώσσα OWL και να εισάγονται στην οντολογία ως στοιχεία της ανάλογα με τους τύπους των ακμών και των κόμβων. Αυτός ο μηχανισμός μπορεί να υποστηρίξει και την εισαγωγή C-OWL σχέσεων, με τη διαφορά ότι οι αλλαγές του στο γράφο θα μετατρέπονται σε C-OWL σχέσεις και θα μεταφέρονται στο αντίστοιχο πλαίσιο C-OWL. Μια πιο μακροπρόθεσμη εξέλιξη του ΕΟΚΟ είναι η δημιουργία ενός έξυπνου πράκτορα μέσα στο πρόγραμμα, που θα αναλύει τις οντολογίες με στόχο να βρίσκει ομοιότητες στη δομή και στο περιεχόμενο τους, ώστε να κάνει προτάσεις για δυνατές σχέσεις C-OWL που μπορούν να δημιουργηθούν. Ένας τέτοιος μηχανισμός χρειάζεται ισχυρούς εσωτερικούς μηχανισμούς δημιουργίας λογικών συλλογισμών, τόσο σε OWL, όσο και σε C-OWL αρχεία.

Βιβλιογραφία

[1] Antoniou, Grigoris, and Frank Van Harmelen. *A semantic web primer*. MIT press, 2004.

[2] McGuinness, Deborah L., and Frank Van Harmelen. "OWL web ontology language overview." *W3C recommendation* 10.2004-03 (2004): 10.

[3] Bouquet, Paolo, et al. "C-owl: Contextualizing ontologies." *The Semantic Web-ISWC 2003*. Springer Berlin Heidelberg, 2003. 164-179.

[4] Vouros, George A., and Georgios Santipantakis. "Combining Ontologies with Correspondences and Link Relations: The E-SHIQ Representation Framework."*arXiv preprint arXiv:1310.2493* (2013).

[5] Baader, Franz, and Werner Nutt. "Basic description logics." *Description logic handbook*. 2003.

[6] http://protege.stanford.edu/

[7] http://semweb.salzburgresearch.at/apps/rdf-gravity/

[8] http://simile.mit.edu/welkin/

[9] http://alignapi.gforge.inria.fr/edoal.html

[10] Scharffe, François, and Dieter Fensel. "Correspondence patterns for ontology alignment." Knowledge Engineering: Practice and Patterns. Springer Berlin Heidelberg, 2008. 83-92.

[11] http://www.w3.org/TR/owl2-overview/

[12] http://owlapi.sourceforge.net/reasoners.html

[13] Kobylinski, Andrzej, and Andrzej Sobczak. "Perspectives in Business Informatics Research."