Διεθνές Πανεπιστήμιο Ελλάδος
Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Μοντελοποίηση Προγράμματος Σπουδών

Φοιτητής:
Ιωάννης Μπόνιος
Αριθμός Μητρώου: 113761

Επιβλέπων:
Αντώνης Σιδηρόπουλος

31 Μαΐου 2025

Τίτλος Π.Ε.: Μοντεολοποίηση Προγράμματος Σπουδών
Κωδικός Π.Ε.: 23189
Ονοματεπώνυμο φοιτητή: Ιωάννης Μπόνιος
Ονοματεπώνυμο εισηγητή: Αντώνης Σιδηρόπουλος
Ημερομηνία ανάληψης Π.Ε.: 29-03-2023
Ημερομηνία περάτωσης Π.Ε.: 31-05-2025

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως πτυχιακή εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε

Abstract

This project focuses on modeling the current curriculum of the department using Semantic Web technologies such as RDF and OWL. The course data, as well as the graduation rules, were extracted from the following sources:
https://www.iee.ihu.gr/udg_courses/ and
https://www.iee.ihu.gr/wp-content/uploads/2024/03/ΜΠΗΣ_Οδηγός_Σπουδών_23-24_v2_.pdf
Specifically, the Protégé platform was used to create the OWL model and SWRL rules, while GraphDB was utilized for running SPARQL queries to extract the required information.
Students of the department can input the codes of the courses they have successfully completed and retrieve the remaining courses required for graduation.
A key advantage of this approach is that it helps reduce the workload of the administrative office, as it automates the answers to questions that students would typically direct to the department staff.

Περίληψη

Η εργασία αυτή εστιάζει στη μοντελοποίηση του τρέχοντος προγράμματος σπουδών του τμήματος, με χρήση τεχνολογιών του Σημασιολογικού Ιστού, όπως RDF και OWL. Τα δεδομένα των μαθημάτων, καθώς και οι κανόνες ολοκλήρωσης σπουδών, έχουν εξαχθεί από τις διευθύνσεις:
https://www.iee.ihu.gr/udg_courses/ και
https://www.iee.ihu.gr/wp-content/uploads/2024/03/ΜΠΗΣ_Οδηγός_Σπουδών_23-24_v2_.pdf
Συγκεκριμένα, χρησιμοποιήθηκαν οι πλατφόρμες Protégé για τη δημιουργία του μοντέλου σε OWL και των κανόνων σε SWRL, καθώς και η GraphDB για την εκτέλεση ερωτημάτων SPARQL και την εξαγωγή των ζητούμενων δεδομένων.
Οι φοιτητές του τμήματος θα μπορούν, εισάγοντας τους κωδικούς των μαθημάτων που έχουν περάσει, να εντοπίζουν τα εναπομείναντα μαθήματα που απαιτούνται για την ολοκλήρωση των σπουδών τους.
Ένα βασικό πλεονέκτημα της προσέγγισης αυτής είναι η αποσυμφόρηση της γραμματείας από ερωτήματα που σχετίζονται με την πρόοδο των φοιτητών, καθώς η πληροφορία αυτή είναι πλέον άμεσα προσβάσιμη.

Περιεχόμενα

Κατάλογος σχημάτων

Κατάλογος πινάκων

Κεφάλαιο 1

Σημασιολογικός Ιστός

Εισαγωγή

Ο Σημασιολογικός Ιστός αντιπροσωπεύει μια όραση για την επόμενη γενιά του Παγκόσμιου Ιστού, όπου οι πληροφορίες δεν είναι μόνο συνδεδεμένες, αλλά και κατανοητές από μηχανές με έναν πιο έξυπνο και ευαίσθητο στο πλαίσιο τρόπο. Η ιδέα είναι να πάμε πέρα από έναν ιστό εγγράφων και υπερσυνδέσμων, ο οποίος είναι κατανοητός από τους ανθρώπους, σε έναν ιστό που είναι επίσης ερμηνεύσιμος από τις μηχανές. Ο Σημασιολογικός Ιστός επιτρέπει σε αυτοματοποιημένα συστήματα, εφαρμογές και λογισμικό να επεξεργάζονται και να αναλύουν δεδομένα με πιο ουσιαστικό και έξυπνο τρόπο, κάνοντάς το τελικά πιο διαδραστικό και αποτελεσματικό.[1]

1.1 Web 1.0

Το Web 1.0 συχνά αποκαλούμενο ως «στατικός ιστός», αντιπροσωπεύει την αρχική φάση του διαδικτύου, από τις αρχές της δεκαετίας του 1990 έως τις αρχές του 2000. Αυτή η εποχή χαρακτηρίστηκε από στατικούς ιστότοπους που λειτουργούσαν κυρίως ως πύλες πληροφόρησης, παρέχοντας περιεχόμενο χωρίς διαδραστικότητα. Οι χρήστες μπορούσαν να διαβάσουν ή να δουν το περιεχόμενο, αλλά δεν είχαν τη δυνατότητα να συμβάλουν, να σχολιάσουν ή να αλληλεπιδράσουν με τις πληροφορίες που παρουσιάζονταν. Οι ιστότοποι εκείνης της περιόδου κατασκευάζονταν με απλή HTML, εστιάζοντας στην παράδοση κειμενοκεντρικού περιεχομένου με ελάχιστη έμφαση στον σχεδιασμό και τη λειτουργικότητα. Η πλοήγηση ήταν γραμμική και βασισμένη σε υπερσυνδέσμους, θυμίζοντας απλούς καταλόγους που επέτρεπαν στους χρήστες να μετακινούνται ανάμεσα σε στατικές σελίδες. Οι ενημερώσεις περιεχομένου ήταν σπάνιες και έπρεπε να γίνονται χειροκίνητα από τους ιδιοκτήτες ή τους διαχειριστές των ιστοτόπων, καθώς δεν υπήρχαν μηχανισμοί για ενημερώσεις σε πραγματικό χρόνο ή δυναμικές αλληλεπι-

δράσεις.
Η έννοια του περιεχομένου που παράγεται από τους χρήστες, χαρακτηριστικό γνώρισμα των επόμενων εξελίξεων του ιστού, απουσίαζε εντελώς, με όλο το υλικό να δημιουργείται και να επιμελείται αποκλειστικά από τους διαχειριστές των σελίδων. Οι ταχύτητες σύνδεσης στο διαδίκτυο εκείνης της περιόδου ήταν αργές, με περιορισμένο εύρος ζώνης που περιόριζε τη χρήση πολυμέσων, όπως εικόνες και βίντεο. Έτσι, οι ιστότοποι ήταν κυρίως κειμενοκεντρικοί και βελτιστοποιημένοι για απόδοση, παρά για αισθητική ή διαδραστικότητα. Επιπλέον, οι μηχανές αναζήτησης της εποχής, όπως η AltaVista, ήταν απλές και λειτουργούσαν περισσότερο σαν κατάλογοι παρά σαν προηγμένα εργαλεία αναζήτησης βασισμένα σε αλγορίθμους. Ο Ιστός 1.0 χρησιμοποιήθηκε κυρίως για τη διανομή στατικού περιεχομένου, όπως ειδησεογραφικά άρθρα, πληροφοριακές σελίδες και εταιρικά προφίλ, χωρίς να ενθαρρύνει διαδραστικές κοινότητες ή επικοινωνία σε πραγματικό χρόνο. Παρά τους περιορισμούς του, αυτή η φάση του ιστού έθεσε τα θεμέλια για τις πιο δυναμικές και συνεργατικές φάσεις που ακολούθησαν. Ο στατικός και αποκλειστικά αναγνωστικός χαρακτήρας του Ιστού 1.0, αν και περιοριστικός για τα σημερινά δεδομένα, έπαιξε ζωτικό ρόλο στην καθιέρωση του διαδικτύου και στη διάθεση πληροφοριών σε παγκόσμια κλίμακα.

1.2 Web 2.0

Το Web 2.0 αποτελεί τη δεύτερη γενιά του Παγκόσμιου Ιστού, μετατρέποντας το διαδίκτυο σε ένα ιδιαίτερα διαδραστικό και προσανατολισμένο στον χρήστη περιβάλλον. Σε αντίθεση με τον στατικό, αναγνωστικό χαρακτήρα του πρώτου, το Web 2.0 δίνει έμφαση στη συμμετοχή των χρηστών, στη συνεργασία και στη δημιουργία περιεχομένου από τους ίδιους τους χρήστες. Αυτή η μετάβαση έφερε στο προσκήνιο δυναμικές πλατφόρμες, όπως τα ιστολόγια, τα wikis και τα μέσα κοινωνικής δικτύωσης, επιτρέποντας στα άτομα να συμβάλλουν ενεργά και να διαμορφώνουν την εμπειρία τους στο διαδίκτυο. Εισήγαγε μια νέα εποχή συνεργασίας και αλληλεπίδρασης, επιτρέποντας στους χρήστες να συνδέονται μέσω πλατφορμών κοινωνικής δικτύωσης όπως το Facebook, το Twitter και το LinkedIn. Αυτός ο συμμετοχικός ιστός προώθησε την κουλτούρα της κοινής γνώσης, μετατρέποντας το διαδίκτυο σε κόμβο επικοινωνίας, δημιουργικότητας και ομαδικής εργασίας.
Ένα από τα καθοριστικά χαρακτηριστικά του είναι η έμφαση στη διαδραστικότητα και την αλληλεπίδραση σε πραγματικό χρόνο. Οι ιστότοποι και οι πλατφόρμες έγιναν πιο δυναμικοί, προσφέροντας ευέλικτες διεπαφές και πλούσιες εμπειρίες χρηστών μέσω πολυμεσικού περιεχομένου, βίντεο και κινούμενων γραφικών. Η άνοδος της κατηγοριοποίησης περιεχομένου μέσω ετικετών επέτρεψε στους χρήστες να οργανώνουν και να βρίσκουν πληροφορίες πιο εύκολα. Επιπλέον, οι υπηρεσίες που βασίζονται στο cloud έφεραν επανάσταση στη διαθεσιμότητα, δίνοντας τη δυνατότητα στους χρήστες να έχουν πρόσβαση σε εφαρμογές και δεδομένα από

οποιοδήποτε σημείο με σύνδεση στο διαδίκτυο. Εργαλεία συνεργασίας, όπως το Google Docs και τα wikis, βελτίωσαν ακόμη περισσότερο την ομαδική εργασία, επιτρέποντας σε πολλούς χρήστες να επεξεργάζονται και να συμβάλλουν ταυτόχρονα.
Ο Ιστός 2.0 διευκόλυνε επίσης την ενσωμάτωση μεταξύ πλατφορμών, επιτρέποντας στις υπηρεσίες να συνεργάζονται και να βελτιώνουν τη λειτουργικότητα. Για παράδειγμα, εφαρμογές όπως το Gmail και το YouTube λειτουργούν απρόσκοπτα μέσα σε προγράμματα περιήγησης, και προσαρμόζονται σε διάφορα μεγέθη οθόνης και συσκευές. Η κοινή χρήση περιεχομένου, οι συνδρομές και οι τακτικές ενημερώσεις έγιναν εύκολες, δημιουργώντας ένα δυναμικό οικοσύστημα για τους χρήστες. Ο κοινωνικός και συμμετοχικός χαρακτήρας του Ιστού 2.0 μετέτρεψε το διαδίκτυο σε έναν συνεργατικό χώρο, όπου οι χρήστες όχι μόνο καταναλώνουν περιεχόμενο αλλά συμμετέχουν ενεργά, το μοιράζονται και διαμορφώνουν τον ψηφιακό κόσμο. Αυτή η εξέλιξη άνοιξε τον δρόμο για τον σύγχρονο ιστό, που βασίζεται στη συνδεσιμότητα και την καινοτομία.[2]

1.3 Προκλήσεις

Ένα συνηθισμένο πρόβλημα είναι η υψηλή ανάκληση(High recall) σε συνδυασμό με χαμηλή ακρίβεια(low precision) όταν χρησιμοποιούμε μηχανές αναζήτησης. Αν και φαίνεται πλεονεκτικό να επιστρέφονται πολλά αποτελέσματα για ένα ερώτημα, αυτό συχνά δημιουργεί μια κατάσταση όπου μόνο ένα μικρό ποσοστό αυτών είναι πραγματικά σχετικό. Ο χρήστης ψάχνει ανάμεσα σε σελίδες με ελάχιστα ή καθόλου σχετικό περιεχόμενο, γεγονός που μειώνει τη χρησιμότητα της αναζήτησης. Αυτός ο υπερβολικός όγκος πληροφοριών μπορεί να καταστήσει την αναζήτηση εξίσου δύσκολη με το να μην υπήρχαν καθόλου αποτελέσματα.
Ένας άλλος περιορισμός είναι η κατακερματισμένη φύση των αποτελεσμάτων αναζήτησης. Όταν οι πληροφορίες είναι διασκορπισμένες σε πολλά έγγραφα, οι μηχανές αναζήτησης συχνά αποτυγχάνουν να τις ενοποιήσουν σε ένα συνεκτικό σύνολο. Οι χρήστες αναγκάζονται να πραγματοποιούν πολλές ξεχωριστές αναζητήσεις, να εξάγουν χειροκίνητα τις επιμέρους πληροφορίες από κάθε αποτέλεσμα και να τις συνδυάζουν μόνοι τους. Αυτή η διαδικασία δεν είναι μόνο χρονοβόρα, αλλά και επιρρεπής σε λάθη, καθώς μπορεί να παραβλεφθούν σημαντικά στοιχεία κατά τη διάρκεια της ολοκλήρωσης.Η διαλειτουργικότητα των δεδομένων αποτελεί επίσης ένα σημαντικό εμπόδιο. Διαφορετικά συστήματα και βάσεις δεδομένων συχνά αποθηκεύουν δεδομένα σε μη συμβατές μορφές, δημιουργώντας εμπόδια στην ενσωμάτωση και συνεργασία. Αυτή η έλλειψη τυποποίησης οδηγεί σε ”σιλό δεδομένων,” όπου πολύτιμες πληροφορίες παραμένουν απομονωμένες μέσα σε συγκεκριμένες πλατφόρμες ή οργανισμούς. Η αδυναμία συνδυασμού και διαμοιρασμού δεδομένων περιορίζει την καινοτομία και εμποδίζει τη λήψη τεκμηριωμένων αποφάσεων.
Η ραγδαία ανάπτυξη του Διαδικτύου έχει επίσης οδηγήσει σε υπερφόρτωση πληροφοριών.

Αν και θεωρητικά είναι πλεονεκτικό να υπάρχει πρόσβαση σε τεράστιο όγκο δεδομένων, η ίδια η ποσότητα μπορεί να γίνει μη διαχειρίσιμη. Οι μηχανές αναζήτησης και άλλα εργαλεία μπορούν να βοηθήσουν τους χρήστες να εντοπίσουν πληροφορίες, αλλά συχνά δεν επαρκούν για να βοηθήσουν τους ανθρώπους ή τις μηχανές να κατανοήσουν το πλαίσιο και τις σχέσεις μεταξύ διαφορετικών δεδομένων. Ως αποτέλεσμα, η τεράστια διαθεσιμότητα πληροφοριών δεν μεταφράζεται πάντα σε ουσιαστική γνώση ή εφαρμόσιμες ιδέες.
Η ασάφεια και η έλλειψη πλαισίου στο διαδικτυακό περιεχόμενο περιπλέκουν περαιτέρω τη διαδικασία ανάκτησης και κατανόησης πληροφοριών. Όροι όπως "μήλο" μπορεί να έχουν πολλαπλές σημασίες ανάλογα με το πλαίσιο, αναφερόμενοι σε ένα φρούτο σε μία περίπτωση και σε μια τεχνολογική εταιρεία σε άλλη. Χωρίς έναν μηχανισμό που να αποσαφηνίζει τέτοιους όρους και να κατανοεί το υποκείμενο πλαίσιο, οι μηχανές δυσκολεύονται να παρέχουν ακριβή και σχετικά αποτελέσματα. Αυτή η ασάφεια μπορεί να οδηγήσει σε παρανοήσεις και αναποτελεσματικότητα, ιδιαίτερα όταν απαιτείται ακριβής κατανόηση.
Τέλος, ο περιορισμένος αυτοματισμός των συστημάτων εξακολουθεί να αποτελεί σημαντικό εμπόδιο. Οι περισσότερες διαδικασίες ερμηνείας και λήψης αποφάσεων βασίζονται σε μεγάλο βαθμό στην ανθρώπινη παρέμβαση, κάτι που είναι χρονοβόρο και επιρρεπές σε λάθη. Παρόλο που έχουν γίνει ορισμένες πρόοδοι στη μηχανική μάθηση και την τεχνητή νοημοσύνη, υπάρχει ακόμα πολύς δρόμος πριν οι μηχανές μπορέσουν να επεξεργαστούν και να αναλύσουν δεδομένα με την ίδια ακρίβεια και ευαισθησία όπως οι άνθρωποι.

1.4 Web 3.0

Η αναπαράσταση του περιεχόμενο του Ιστού σε μια μορφή που να είναι πιο εύκολα επεξεργάσιμη από μηχανές και να χρησιμοποιεί ευφυείς τεχνικές για να εκμεταλλευτεί αυτές τις αναπαραστάσεις, αναφερόμαστε σε αυτό το σχέδιο επανάστασης του Ιστού ως Σημασιολογικός Ιστός. Είναι σημαντικό να κατανοήσουμε ότι ο Σημασιολογικός Ιστός δεν θα είναι ένα νέο παγκόσμιο πληροφοριακό σύστημα παράλληλο με τον υπάρχοντα Παγκόσμιο Ιστό· αντίθετα, θα εξελιχθεί σταδιακά μέσα από τον υπάρχοντα Ιστό.
Η έννοια του Σημασιολογικού Ιστού εισήχθη για πρώτη φορά από τον Tim Berners-Lee, τον εφευρέτη του Παγκόσμιου Ιστού, στα τέλη της δεκαετίας του 1990. Ο Berners-Lee πρότεινε την ιδέα ενός «ιστού δεδομένων» ως έναν τρόπο να επιτρέψει στις μηχανές να κατανοούν και να ερμηνεύουν τα δεδομένα αυτόματα, ενισχύοντας τη σημερινή δομή του Ιστού. Ανέγνωσε ότι, ενώ τα έγγραφα HTML ήταν εξαιρετικά για ανθρώπινη κατανόηση, ο Ιστός δεν είχε τη δυνατότητα να υποστηρίξει τις μηχανές στην εξαγωγή ουσιαστικών συμπερασμάτων από τα δεδομένα.
Το 2001, η Κοινοπραξία του Παγκοσμίου Ιστού W3C (World Wide Web Consortium) τυποποίησε την όραση για το Σεμαντικό Διαδίκτυο και ξεκίνησε την ανάπτυξη προδιαγραφών και προτύ-

πων για την υλοποίηση αυτής της ιδέας. Αυτά τα πρότυπα επικεντρώθηκαν στην ανάπτυξη μιας πιο δομημένης, αναγνώσιμης από μηχανές μορφής για τα δεδομένα και είχαν στόχο να αντιμετωπίσουν τις προκλήσεις της διαχείρισης όλο και μεγαλύτερων και πιο σύνθετων συνόλων δεδομένων με διασυνδεδεμένο τρόπο. Η ανάπτυξη του Σημασιολογικού Ιστού έχει αποκτήσει μεγάλη δυναμική στη βιομηχανία, ενώ οι κυβερνήσεις επενδύουν σημαντικά σε αυτό με την κυβέρνηση των Ηνωμένων Πολιτειών να ιδρύει το έργο DAML (DARPA Agent Markup Language).

1.5 Τρόποι αντιμετώπισης

Για την αντιμετώπιση των σημαντικών προκλήσεων που προκύπτουν από το παραδοσιακό διαδίκτυο, γίνεται χρήση καινοτόμων τεχνολογιών που επιτρέπουν στις μηχανές να κατανοούν καλύτερα τα δεδομένα και να παρέχουν πιο ακριβείς και χρήσιμες πληροφορίες. Η βασική προσέγγιση στηρίζεται στη δομή και την επισήμανση των δεδομένων με τρόπο που να είναι κατανοητός από τις μηχανές.

Ένας από τους βασικούς τρόπους με τους οποίους ο Σημασιολογικός Ιστός επιλύει τα προβλήματα είναι μέσω της διαλειτουργικότητας των δεδομένων. Χρησιμοποιώντας τυποποιημένα μοντέλα, όπως το Resource Description Framework (RDF), τα δεδομένα οργανώνονται σε τριπλέτες που αποτελούνται από υποκείμενο, ρήμα και αντικείμενο. Αυτές οι τριπλέτες περιγράφουν σχέσεις ή «αλήθειες» για τα δεδομένα, παρέχοντας ένα συνεκτικό πλαίσιο που διευκολύνει την ανταλλαγή και την κοινή χρήση πληροφοριών μεταξύ διαφορετικών συστημάτων. Με αυτόν τον τρόπο, τα δεδομένα γίνονται προσβάσιμα και αξιοποιήσιμα σε μεγαλύτερη κλίμακα.

Μια άλλη σημαντική προσέγγιση είναι η παροχή πλαισίου μέσα από τη χρήση οντολογιών. Οι οντολογίες είναι τυποποιημένες αναπαραστάσεις γνώσης που ορίζουν τις σχέσεις μεταξύ εννοιών σε έναν συγκεκριμένο τομέα. Για παράδειγμα, μπορούν να διευκρινίσουν αν ο όρος ”apple” αναφέρεται σε φρούτο ή σε μια τεχνολογική εταιρεία, ανάλογα με το πλαίσιο. Αυτή η προσέγγιση βοηθά στην επίλυση της ασάφειας και στη διασφάλιση ότι οι μηχανές κατανοούν με ακρίβεια τη σημασία των δεδομένων.

Η αυτοματοποίηση είναι ένας άλλος κρίσιμος παράγοντας που καθιστά το Σημασιολογικό Ιστό μοναδικό. Με την οργάνωση των δεδομένων σε μορφή αναγνώσιμη από μηχανές, τα αυτοματοποιημένα συστήματα μπορούν να εξάγουν νέες πληροφορίες από τα υπάρχοντα δεδομένα, να διενεργούν πολύπλοκες αναζητήσεις και να προσφέρουν πιο εξειδικευμένες και ακριβείς συστάσεις. Αυτό μειώνει την εξάρτηση από τον ανθρώπινο παράγοντα και επιταχύνει τις διαδικασίες λήψης αποφάσεων.

Επίσης, οι μηχανές αναζήτησης που χρησιμοποιούν σημασιολογικές τεχνολογίες μπορούν να ερμηνεύουν την πρόθεση του χρήστη και να παρέχουν αποτελέσματα που είναι πιο σχετικά με τις πραγματικές ανάγκες του. Τέλος, οι συνδέσεις μεταξύ δεδομένων γίνονται πιο δυναμικές

και πλούσιες, διευκολύνοντας την ανακάλυψη νέων σχέσεων και μοτίβων. Ως αποτέλεσμα την σημαντική βελτίωση της διαδικασίας αναζήτησης και ανακάλυψης.
Με αυτές τις προσεγγίσεις, ο Σημασιολογικός Ιστός προσφέρει λύσεις που καθιστούν την πρόσβαση και τη χρήση των πληροφοριών πιο ακριβή, αποτελεσματική και ευφυή, φέρνοντας το όραμα ενός καλύτερα συνδεδεμένου κόσμου πιο κοντά στην πραγματικότητα.

1.6 Εφαρμογές

Data Integration in Oil & Gas – Chevron: Για πολλά χρόνια, η Chevron πειραματίζεται με τεχνολογίες Σημασιολογικού Ιστού σε μια σειρά εφαρμογών. Οι τεχνολογίες του Σημασιολογικού Ιστού επιτρέπουν στους μηχανικούς και τους ερευνητές να συνδυάζουν αυθαίρετα δεδομένα με αυθαίρετους τρόπους σε μια προσπάθεια να κατανοήσουν καλύτερα και να προβλέψουν τις καθημερινές λειτουργίες των κοιτασμάτων πετρελαίου.

Αναζήτηση στον Ιστό και ηλεκτρονικό εμπόριο: Οι μηχανές αναζήτησης επωφελούνται πραγματικά από την πρόσβαση σε επιπλέον μεταδεδομένα προκειμένου να αποδίδουν πιο σχετικά αποτελέσματα. Οι μεγαλύτερες εταιρείες του χώρου επενδύουν πολλά σε πρότυπα που ενθαρρύνουν τις επιχειρήσεις να σχολιάζουν τις ιστοσελίδες τους με σημαντικά μεγαλύτερη δομή, κάτι που ήταν μια από τις αρχικές προθέσεις του οράματος του Σημασιολογικού Ιστού. Το ίδιο το RDF μπορεί ακόμη και να ενσωματωθεί σε ιστοσελίδες μέσω RDFa.

Media Management – BBC: Με διαφορά η πιο ευρεία δημόσια χρήση των τεχνολογιών του Σημασιολογικού Ιστού είναι ο ιστότοπος του BBC. Το 2010, ολόκληρη η ιστοσελίδα τους για το Παγκόσμιο Κύπελλο τροφοδοτήθηκε από τεχνολογίες Σημασιολογικού Ιστού, όπως αναφέρθηκε στο ReadWriteWeb και το SemanticWeb.com. Ακόμη και σήμερα, μεγάλα τμήματα του δημόσιου ιστότοπού τους εκτελούνται σε τεχνολογίες Σημασιολογικού Ιστού. Επίσης: Time Inc., η Elsevier και η Βιβλιοθήκη του Κογκρέσου διαθέτουν συστήματα παραγωγής που έχουν κατασκευαστεί χρησιμοποιώντας τεχνολογίες Σημασιολογικού Ιστού.

Supply Chain Management – Biogen Idec: Φαρμακευτική εταιρεία γνωστή για την παραγωγή φαρμάκων που χρησιμοποιούνται για τη θεραπεία της σκλήρυνσης κατά πλάκας. Διαχειρίζεται την παγκόσμια αλυσίδα εφοδιασμού της χρησιμοποιώντας τεχνολογίες Σημασιολογικού Ιστού. Ως κατηγορία προβλημάτων, η διαχείριση της εφοδιαστικής αλυσίδας περιλαμβάνει πολλά χαρακτηριστικά που την καθιστούν ώριμη για την εφαρμογή Τεχνολογιών Σημασιολογικού Ιστού. [3]

Κεφάλαιο 2

Οντολογίες και Λεξιλόγια

Εισαγωγή

Η έννοια της «οντολογίας» στην επιστήμη των υπολογιστών προέρχεται από τη φιλοσοφία, όπου έχει μακρά ιστορία. Οι φιλόσοφοι επηρεάζουν την ανάπτυξη των οντολογιών, καθώς απαιτείται προσεκτική εννοιολόγηση. Ένα βασικό ζήτημα είναι αν μια οντολογία αναπαριστά την πραγματικότητα ή μια εννοιολογική σύλληψη. Στους επιστημονικούς τομείς, οι οντολογίες πρέπει να ευθυγραμμίζονται με την πραγματικότητα για να αποφευχθούν λάθη, ενώ αλλού αρκεί η εννοιολόγηση. Οι εμπειριστές θεωρούν ότι οι επιστημονικοί όροι αναφέρονται σε υπαρκτές οντότητες, ενώ οι εννοιοκράτες τους βλέπουν ως νοητικές κατασκευές. Ένα άλλο ζήτημα αφορά τα καθολικά—αν οι γενικοί επιστημονικοί όροι αναφέρονται σε ανεξάρτητες οντότητες ή αφηρημένες έννοιες. Αν και τέτοιες φιλοσοφικές έρευνες μπορούν να επηρεάσουν τον σχεδιασμό των οντολογιών, δεν είναι πάντα απαραίτητες, ιδιαίτερα σε πρακτικές εφαρμογές. Η φιλοσοφία βοηθά στη διάκριση εννοιών στη μοντελοποίηση, όπως η διαφορά μεταξύ ρόλων και οντοτήτων ή γεγονότων και συμμετοχής, που επηρεάζουν τη δομή της γνώσης στις οντολογίες.

2.1 Ορισμός Οντολογίας

Για να μπορέσουμε να ορίσουμε τι είναι μια οντολογία, θα τη συγκρίνουμε πρώτα με κάποιες τεχνολογίες με τις οποίες είμαστε ήδη εξοικειωμένοι όπως σχεσιακές βάσεις δεδομένων και εννοιολογικά μοντέλα δεδομένων πχ EER,UML.
Μια σημαντική διάκριση μεταξύ εννοιολογικών μοντέλων δεδομένων και οντολογιών είναι ότι τα εννοιολογικά μοντέλα δεδομένων σχεδιάζονται για να αναπαριστούν δεδομένα ειδικά για μια συγκεκριμένη εφαρμογή. Εστιάζουν στο πώς θα δομηθούν και θα χρησιμοποιηθούν τα δεδομένα μέσα σε αυτό το σύστημα, ενώ παραμένουν ανεξάρτητα από μία μελλοντική εφαρμογή.

Οι οντολογίες παρέχουν μια ευρύτερη, ανεξάρτητη από εφαρμογές αναπαράσταση γνώσης για έναν συγκεκριμένο τομέα (Domain). Αυτό σημαίνει ότι οι οντολογίες δεν συνδέονται με καμία συγκεκριμένη εφαρμογή αλλά σχεδιάζονται ώστε να είναι επαναχρησιμοποιήσιμες σε πολλά συστήματα και πλαίσια.
Από αυτή τη διάκριση προκύπτουν περαιτέρω διαφορές σχετικά με το περιεχόμενό τους, το σκοπό και το εύρος τους. Οι οντολογίες είναι συνήθως τυποποιημένες σε μια γλώσσα λογικής, ενώ η εννοιολογική μοντελοποίηση αφορά περισσότερο τη σχεδίαση πλαισίων και γραμμών με άτυπο τρόπο, και χρησιμοποιούνται διαφορετικά και εξυπηρετούν διαφορετικούς σκοπούς.
Οι σχεσιακές βάσεις δεδομένων (RDBMS) και οι οντολογίες ως βάσεις γνώσης (knowledge bases) διαφέρουν σημαντικά στον τρόπο με τον οποίο διαχειρίζονται και αναπαριστούν τη γνώση. Ενώ οι RDBMS επικεντρώνονται κυρίως στην αποθήκευση και την ανάκτηση δομημένων δεδομένων με απλά ερωτήματα, οι οντολογίες περιλαμβάνουν την αναπαράσταση της γνώσης ρητά, με χρήση κανόνων και αυτόματη συλλογιστική για την εξαγωγή έμμεσης γνώσης και την ανίχνευση ασυνέπειας.Καθώς επίσης λειτουργούν υπό την Υπόθεση Ανοιχτού Κόσμου (Open World Assumption) σε αντίθεση με το CWA.
Αν και έχουν γίνει αρκετές προσπάθειες απόδοσης ορισμού για το τι είναι οντολογία δεν υπάρχει κάποιος ομόφωνα συμφωνημένος ορισμός.
Στο έγγραφο ορόσημο του Guarino Nicola δίνεται ένας περιεκτικός ορισμός:

An ontology is a logical theory accounting for the intended meaning of a formal vocabulary, i.e. its ontological commitment to a particular conceptualization of the world. The intended models of a logical language using such a vocabulary are constrained by its ontological commitment. An ontology indirectly reflects this commitment (and the underlying conceptualization) by approximating these intended models.

Ενώ έχουν δωθεί και άλλοι απλούστεροι όπως αυτός από τους προγραμματιστές του World Wide Web Consortium (W3C):

An ontology being equivalent to a Description Logic knowledge base.

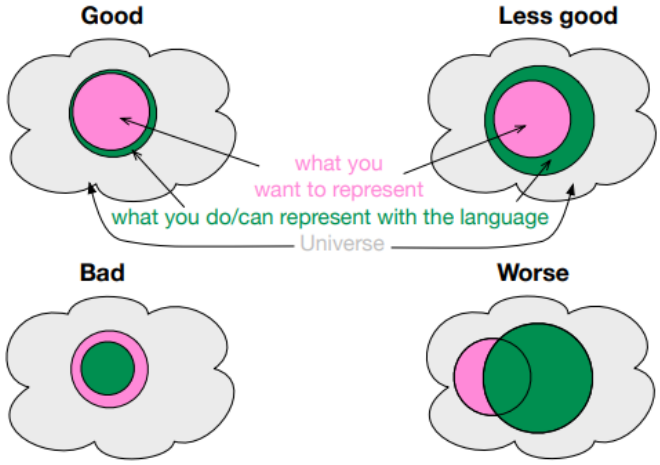
2.2 Καλές και κακές Οντολογίες

Όπως κάποιος μπορεί να γράψει καλό και κακό κώδικα, μπορεί να φτιάξει καλές και κακές οντολογίες. Το πως θα τις χαρακτηρίσουμε όμως είναι λίγο πιο περίπλοκο από ό,τι με τον κώδικα λογισμικού. Στο λογισμικό, ο κακός κώδικας μπορεί να μη μεταγλωττίζεται, να έχει σφάλματα ή να είναι δύσκολος στη συντήρηση. Για τις οντολογίες, το αντίστοιχο του «δεν μεταγλωττίζεται» είναι μια παραβίαση της σύνταξης, που καθιστά την οντολογία άκυρη. Αντίστοιχα, οι οντολογίες μπορεί να έχουν δύο είδη «σφαλμάτων». Το πρώτο είναι λογικά σφάλ-

ματα, όπου αντικρουόμενοι περιορισμοί καθιστούν μια κλάση μη χρησιμοποιήσιμη (π.χ., μια κλάση να μην έχει καθόλου στιγμιότυπα). Το δεύτερο αφορά σημασιολογικά σφάλματα, όπου η οντολογία είναι λογικά σωστή αλλά αντιπροσωπεύει κάτι μη επιθυμητό. Αυτά τα σφάλματα επηρεάζουν την ακρίβεια της αναπαράστασης της οντολογίας.

Πέρα από τη βασική ορθότητα, οι καλές οντολογίες στηρίζονται και σε σωστές δομικές επιλογές. Για παράδειγμα, η έννοια των «πράσινων μήλων» μπορεί να μοντελοποιηθεί είτε ως «μήλα με την ιδιότητα πράσινο» είτε ως «πράσινα αντικείμενα με σχήμα μήλου». Παρόλο που λογικά είναι ισοδύναμα, το πρώτο είναι οντολογικά προτιμότερο, καθώς το ”Μήλο” είναι καταλληλότερο για ταξινόμιση σε σχέση με το ”Πράσινο” που λειτουργεί καλύτερα ως ιδιότητα.

Μια καλή οντολογία ισορροπεί μεταξύ ακρίβειας και κάλυψης. Η ακρίβεια αφορά το πόσο πιστά αντιπροσωπεύει αυτό που προορίζεται, χωρίς να περιλαμβάνει περιττές έννοιες. Η κάλυψη αναφέρεται στο πόσο ολοκληρωμένα καλύπτει τον προοριζόμενο τομέα. Μια καλή οντολογία έχει υψηλή ακρίβεια και μέγιστη κάλυψη, διασφαλίζοντας ότι όλες οι σχετικές έννοιες εκπροσωπούνται με ακρίβεια. Τα προβλήματα εμφανίζονται όταν η ακρίβεια ή η κάλυψη είναι χαμηλή. Για παράδειγμα, η χαμηλή ακρίβεια με μέγιστη κάλυψη περιλαμβάνει άσχετες έννοιες, ενώ η υψηλή ακρίβεια με περιορισμένη κάλυψη παραλείπει κρίσιμες έννοιες. Το χειρότερο σενάριο συνδυάζει χαμηλή ακρίβεια και περιορισμένη κάλυψη, καθιστώντας την οντολογία άχρηστη. Η σωστή ισορροπία απαιτεί προσεκτική μοντελοποίηση και επιλογή της κατάλληλης λογικής αναπαράστασης.[4]



Σχήμα 2.1: Καλές και κακές Οντολογίες

2.3 Λογική πρώτης τάξης

Η Λογική Πρώτης Τάξης (First-Order Logic - FOL) είναι ένα επίσημο σύστημα που επεκτείνει τη λογική προτάσεων (propositional language), επιτρέποντας την αναπαράσταση πιο σύνθετης γνώσης με μεγαλύτερη ακρίβεια. Χρησιμοποιείται για τη μελέτη των σχέσεων μεταξύ της αλήθειας δηλώσεων, αντί της ίδιας της αλήθειας τους στην πραγματικότητα. Δηλαδή σε μία δήλωση “If angels exist then necessarily all of them fit on the head of a needle” εξετάζει στην περίπτωση που το ‘ if ’ μέρος είναι αληθές πώς σχετίζεται με το αληθές μέρος του ‘ then ‘.

Τα βασικά στοιχεία της FOL είναι:

Η σύνταξη: Υπαρξη κατηγορημάτων (predicates) τα οποία χωρίζονται σε μονομελή αυτά δηλαδή που έχουν ένα όρισμα C(a) και τα διμελή(binary) που δέχονται δύο ορίσματα S(a,b) δηλώνουμε δηλαδή ότι μεταξύ των ατόμων a και b υφίσταται η σχέση που εκφράζει το κατηγόρημα S χωρίς να ισχύει όμως η συνεπαγωγή S(b,a).
Χρήση των ποσοδεικτών (quantifiers) για την απόδοση ολοκληρωμένου νοήματος στις εκφράσεις. Υπάρχουν δύο είδη ποσοδεικτών ο καθολικός (universal) ποσοδείκτης, που συμβολίζεται με ∀ και διαβάζεται “για κάθε” και ο υπαρξιακός (existential) ποσοδείκτης, που συμβολίζεται με ∃ και διαβάζεται “υπάρχει έστω ένα”.
Σε εκφράσεις χωρίς ποσοδείκτες όπως C(x) η μεταβλητή x εμφανίζεται ως ελεύθερη (free),
ενώ στις εκφράσεις ∀xC(x) και ∃xC(x) η μεταβλητή x είναι πλέον δεσμευμένη. Τέτοιες εκφράσεις με πλήρες νόημα, όπου όλες οι μεταβλητές που εμφανίζονται είναι δεσμευμένες, ονομάζονται προτάσεις (sentences)
Χρήση συμβόλων όπως συνδέσμους (¬, →, ↔, ∧, ∨, (and)), μεταβλητών, σταθερές, συναρτήσεις και σχέσεις ορίζει τους κανόνες για τη δημιουργία έγκυρων προτάσεων Αυτό επιτρέπει προτάσεις της φυσικής γλώσσας να τυποποιούνται σε λογικές εκφράσεις.

Παράδειγμα:
φυσική γλώσσα Each animal is an organism
λογική πρώτης τάξης ∀x(Animal(x) → Organism(x))

Η σημασιολογία αποδίδει νόημα σε αυτές τις προτάσεις, ερμηνεύοντάς τες μέσα σε δομές που αποτελούνται από ένα υποκείμενο σύνολο και αντιστοιχίσεις για σταθερές, συναρτήσεις και σχέσεις. Οι προτάσεις στη FOL είναι τύποι χωρίς ελεύθερες μεταβλητές, ενώ μια θεωρία είναι ένα συνεπές σύνολο τέτοιων προτάσεων. Τα μοντέλα είναι δομές που ικανοποιούν τις προτάσεις μιας θεωρίας. Η FOL εφαρμόζεται ευρέως σε περιοχές όπως η τυποποίηση διαγραμμάτων (π.χ. UML) ή η ανασύνθεση οντολογιών, όπου οι γραφικές αναπαραστάσεις μετατρέπονται σε ακριβείς λογικούς αξιωματικούς κανόνες. Η αλληλεπίδραση μεταξύ μοντέλων και θεωριών εί-

ναι κεντρική για την κατανόηση της συνέπειας, της πληρότητας και της λογικής συμπερασματολογίας, καθιστώντας τη FOL ένα ισχυρό εργαλείο για την αυστηρή και σαφή συλλογιστική γύρω από τη ρητή και άρρητη γνώση.

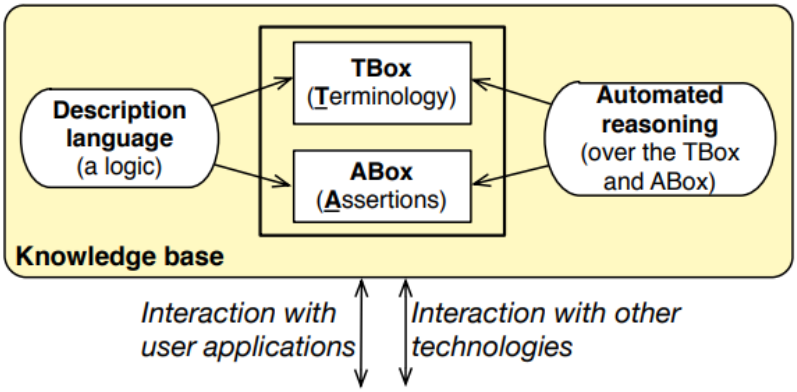
2.4 Αυτοματοποιημένη συλλογιστική

Η αυτοματοποιημένη συλλογιστική περιλαμβάνει τη χρήση τυπικών γλωσσών λογικής και αλγορίθμων για να συμπεράνει σιωπηρή γνώση από ένα σύνολο προϋποθέσεων, επιτρέποντας την κλιμάκωση και την αποδοτική επίλυση προβλημάτων σε σύγκριση με τις χειροκίνητες μεθόδους. Περιλαμβάνει τεχνικές όπως η απαγωγή (deduction), η εικασία (abduction) και η επαγωγή (induction), καθεμία με διαφορετικό σκοπό: η απαγωγή εξάγει συμπεράσματα που προκύπτουν λογικά από μια θεωρία, η εικασία δημιουργεί πιθανές εξηγήσεις για παρατηρήσεις, και η επαγωγή γενικεύει από συγκεκριμένα παραδείγματα σε ευρύτερα συμπεράσματα, αν και με πιθανές ανακρίβειες. Η αυτοματοποιημένη συλλογιστική έχει πρακτικές εφαρμογές σε τομείς όπως η επαλήθευση υλικού, ο προγραμματισμός και η επιστημονική ανακάλυψη. Ωστόσο, η αποτελεσματικότητά της περιορίζεται από την υπολογιστική πολυπλοκότητα των γλωσσών αναπαράστασης, οδηγώντας σε αντισταθμίσεις μεταξύ εκφραστικότητας και αποδοτικότητας. Βασικές παράμετροι στην αυτοματοποιημένη συλλογιστική περιλαμβάνουν η επιλογή της κλάσης προβλημάτων, της τυπικής γλώσσας, της μεθόδου συλλογισμού (π.χ. πίνακες ή ανάλυση) και στρατηγικές βελτιστοποίησης για να διασφαλιστεί η ορθότητα και η πληρότητα. [5]

2.5 Περιγραφικές Λογικές

Μια Λογική Περιγραφής (Description Logic - DL) είναι ένα δομημένο τμήμα της Κατηγορηματικής Λογικής Πρώτης Τάξης (First-Order Logic - FOL). Η αναπαράστασή της βρίσκεται στο επίπεδο των κατηγορημάτων, δεν υπάρχουν μεταβλητές στο τυπικό σύστημα. Οι DL παρέχουν μια λογική ανακατασκευή και μια ενοποιητική τυπική προσέγγιση για άλλες γλώσσες αναπαράστασης γνώσης, όπως συστήματα βασισμένα σε πλαίσια (frames), αντικειμενοστρεφή μοντελοποίηση, σημασιολογικά μοντέλα δεδομένων κ.ά. Παρέχουν τη γλώσσα για τη διατύπωση θεωριών και συστημάτων που εκφράζουν δηλωτικά δομημένη γνώση και τη συλλογιστική με αυτήν. Χρησιμοποιούνται, μεταξύ άλλων, για ορολογίες και οντολογίες, λογική εννοιολογική μοντελοποίηση δεδομένων και ενοποίηση πληροφοριών. [6]

Αξιώματα Ορολογίας (TBox):
Τα αξιώματα ορολογίας ορίζουν σχέσεις μεταξύ εννοιών, επιτρέποντας τη διατύπωση γενικών κανόνων. Οι βασικές μορφές είναι η υπαγωγή (C⊑DC \sqsubseteq DC⊑D), που δηλώνει ότι κάθε στιγμιότυπο της έννοιας CCC ανήκει και στην έννοια DDD, και η ισοδυναμία (A≡CA \equiv



Σχήμα 2.2: Η βασική επισκόπηση των κύριων συστατικών μιας βάσης γνώσης DL.

CA≡C), που εκφράζει ότι δύο έννοιες περιγράφουν το ίδιο σύνολο αντικειμένων. Τα αξιώματα του TBox χρησιμοποιούνται για την οργάνωση και ιεράρχηση γνώσης, εξασφαλίζοντας συνοχή στο σύστημα.

Αξιώματα Ισχυρισμών (ABox)
Τα αξιώματα ισχυρισμών περιγράφουν συγκεκριμένα αντικείμενα και τις σχέσεις τους. Περιλαμβάνουν δηλώσεις του τύπου C(a) (το άτομο a ανήκει στην έννοια C) και r(a,b) (τα άτομα a και b σχετίζονται μέσω του ρόλου r). Επιπλέον, μπορούν να εκφράσουν ανισότητες μεταξύ ατόμων. Τα αξιώματα του ABox επιτρέπουν την αναπαράσταση συγκεκριμένων δεδομένων και υποστηρίζουν συλλογιστική για την εξαγωγή νέας γνώσης.

Η σημασιολογία της Περιγραφικής Λογικής (DL) καθορίζει τη σημασία των αξιωμάτων και τις λογικές συνέπειες μιας οντολογίας. Βασίζεται στην Αρχή του Ανοικτού Κόσμου, δηλαδή η έλλειψη πληροφορίας δεν συνεπάγεται με το ψεύδος(false) αλλά άγνοια, επιτρέποντας πολλαπλές δυνατές ερμηνείες. Όσο περισσότερα αξιώματα περιλαμβάνονται, τόσο αυστηρότεροι είναι οι περιορισμοί και λιγότερες οι επιτρεπτές ερμηνείες. Οι DLs δεν υιοθετούν την υπόθεση μοναδικότητας ονομάτων, επιτρέποντας διαφορετικά ονόματα να αναφέρονται στο ίδιο αντικείμενο και να υπάρχουν άγνωστα άτομα στο πεδίο ορισμού.Η υπολογιστική επεξεργασία των συνεπαγωγών απαιτεί εξειδικευμένους αλγορίθμους, καθώς είναι αδύνατο να ελεγχθούν όλες οι πιθανές ερμηνείες. Υπάρχουν εργαλεία που ελέγχουν συνέπεια, υπαγωγή εννοιών και περιπτώσεις ατόμων, χρησιμοποιώντας αποδοτικούς αλγορίθμους λογικού συμπερασμού.
Οι Περιγραφικές Λογικές μάς ενδιαφέρουν γιατί η γλώσσα OWL βασίζεται σε αυτές και έχουν το πλεονέκτημα οτι είναι αποκρίσιμες (decidable). Αυτό πρακτικά σημαίνει ότι αν έχουμε έναν ισχυρισμό (πρόταση), τότε μπορούμε να διαπιστώσουμε αν αληθεύει ή όχι με αλγοριθμικό, δηλαδή με αυτόματο τρόπο. Οι αλγόριθμοι που έχουν αναπτυχθεί στο πλαίσιο των Περιγραφικών Λογικών μπορούν να χρησιμοποιηθούν από το ειδικό λογισμικό που διαχειρίζεται τις οντολο-

γίες της OWL. Με αυτό τον τρόπο το λογισμικό «γνωρίζει» τη σημασιολογία της οντολογίας και μπορεί να εξάγει συμπεράσματα από αυτή. [7]

2.5.1 ALC

Η Περιγραφική Λογική ALC (Attributive Language with Complement) είναι μια από τις απλούστερες αλλά χρήσιμες Περιγραφικές Λογικές (ΠΛ). Ανήκει σε μια οικογένεια λογικών που χρησιμοποιούνται για την αναπαράσταση γνώσης και την αυτοματοποιημένη συλλογιστική. Η ALC προτάθηκε το 1991 και η ονομασία της προέρχεται από τα αρχικά των λέξεων "Attributive Language with Complement", καθώς είναι μια επέκταση της βασικής γλώσσας AL με την προσθήκη της άρνησης (complement).

Η ALC βασίζεται σε τρία βασικά σύνολα:
Ονόματα εννοιών (NC): Αναπαριστούν έννοιες (π.χ., "Άνθρωπος", "Ζώο").
Ονόματα ρόλων (NR): Αναπαριστούν σχέσεις μεταξύ εννοιών (π.χ., "έχει γονέα").
Ατομικά ονόματα (NI): Αναπαριστούν συγκεκριμένα αντικείμενα (π.χ., "Ο Γιάννης").

Οι έννοιες (concepts) στην ALC ορίζονται
Κάθε όνομα έννοιας $A \in NC$ είναι ALC έννοια.
Αν C και D είναι ALC έννοιες και $r \in NR$, τότε οι παρακάτω εκφράσεις είναι επίσης ALC έννοιες:
Σύζευξη: $C \sqcap D$ (και τα δύο ισχύουν).
Δ**ιάζευξη**: $C \sqcup D$ (τουλάχιστον ένα ισχύει).
Άρνηση: $\neg C$ (όχι C).
Υπαρξιακός περιορισμός: $\exists r.C$ (υπάρχει κάποιος που σχετίζεται με r και ικανοποιεί την έννοια C).
Κ**αθολικός περιορισμός**: $\forall r.C$ (όλοι όσοι σχετίζονται με r ικανοποιούν την έννοια C).

Επίσης, χρησιμοποιούνται τα σύμβολα:
$\top$ (top) για την έκφραση $A \sqcup \neg A$ (πάντα αληθές).
$\bot$ (bottom) για την έκφραση $A \sqcap \neg A$ (πάντα ψευδές).

2.5.2 SROIQ

Η Περιγραφική Λογική SROIQ αποτελεί το λογικό υπόβαθρο της γλώσσας OWL και διακρίνεται για τη μεγάλη εκφραστική της δύναμη, παραμένοντας ωστόσο αποκρίσιμη. Το όνομά της προκύπτει από επεκτάσεις της βασικής λογικής ALC, ενσωματώνοντας χαρακτηριστικά όπως μεταβατικότητα ρόλων (S), ιεραρχίες ρόλων (H), αξιώματα ρόλων (R), ονοματικές έννοιες (O), αντίστροφους ρόλους (I) και αριθμητικούς περιορισμούς (Q).

Η σύνταξη της SROIQ περιλαμβάνει έννοιες όπως σύζευξη (⊓), διάζευξη (⊔), άρνηση (¬), υπαρξιακούς (∃r.C) και καθολικούς (∀r.C) περιορισμούς, καθώς και αριθμητικούς περιορισμούς (≥ n r.C, ≤ n r.C). Επιπλέον, υποστηρίζει ονοματικές έννοιες, που επιτρέπουν τον ορισμό εννοιών μέσω ατομικών ονομάτων.
Η σημασιολογία της βασίζεται σε ερμηνείες που συσχετίζουν έννοιες με υποσύνολα ενός πεδίου, ρόλους με σχέσεις και ατομικά ονόματα με συγκεκριμένα στοιχεία. Διατηρεί την αυξημένη λειτουργικότητα των ρόλων, περιλαμβάνοντας καθολικούς ρόλους (u), αντίστροφους ρόλους (r^-) και αυτοπεριορισμούς (∃r.Self).
Η SROIQ αποτελεί επέκταση της ALC, ενισχύοντας τη δύναμη της περιγραφικής λογικής και επιτρέποντας πιο σύνθετους συλλογισμούς σε συστήματα που βασίζονται στην OWL.

2.6 Δημιουργία οντολογιών και λεξιλογίων

Μια οντολογία είναι ουσιαστικά μια επίσημη αναπαράσταση της γνώσης σε έναν συγκεκριμένο τομέα. Ορίζει ένα σύνολο εννοιών και κατηγοριών, μαζί με τις ιδιότητές τους και τις μεταξύ τους σχέσεις. Παρέχοντας μια δομημένη και τυποποιημένη ερμηνεία ενός τομέα γνώσης, οι οντολογίες επιτρέπουν στα συστήματα σημασιολογικής αναζήτησης να κατανοούν το πλαίσιο και να εξάγουν ουσιαστικά συμπεράσματα.Δομούνται ως ένα ιεραρχικό δίκτυο κλάσεων και υποκλάσεων που αντιπροσωπεύουν έννοιες μέσα σε έναν τομέα. Κάθε κλάση μπορεί να έχει ιδιότητες (χαρακτηριστικά) και μπορεί να συνδεθεί με άλλες κλάσεις μέσω σχέσεων. Η πιο θεμελιώδης σχέση είναι η σχέση ”is-a”, που αντιπροσωπεύει την κληρονομικότητα μεταξύ μιας κλάσης και της υποκατηγορίας της.Επιπλέον, οι οντολογίες μπορούν να περιλαμβάνουν στιγμιότυπα ή συγκεκριμένα παραδείγματα κλάσεων
Ενώ οι οντολογίες παρέχουν δομή, τα λεξιλόγια παρέχουν μια γλώσσα. Είναι σύνολα όρων ή λέξεων που αφορούν έναν συγκεκριμένο τομέα, μαζί με ορισμούς που διευκρινίζουν τη σημασία και τη χρήση αυτών των όρων. Βοηθούν στην αποσαφήνιση των όρων και διασφαλίζουν τη συνεπή κατανόηση μεταξύ διαφορετικών συστημάτων.Η δομή ενός λεξιλογίου είναι κάπως πιο απλή από μια οντολογία. Κάθε όρος στο λεξιλόγιο αντιπροσωπεύει μια έννοια και ορίζεται με τρόπο που αντανακλά το νόημά του εντός του συγκεκριμένου τομέα ενδιαφέροντος. Οι ίδιοι οι ορισμοί μπορούν να σχολιαστούν με πληροφορίες σχετικά με τη χρήση του όρου σε διαφορετικά πλαίσια.Μαζί, οι οντολογίες και τα λεξιλόγια επιτρέπουν στα συστήματα να υπερβαίνουν την απλή αντιστοίχιση λέξεων-κλειδιών και να παρέχουν αποτελέσματα που είναι πραγματικά σχετικά με την πρόθεση και το πλαίσιο του χρήστη. Δημιουργώντας πλούσιους, διασυνδεδεμένους ιστούς νοήματος, επιτρέπουν το «σημασιολογικό» μέρος της σημασιολογικής αναζήτησης. Για παράδειγμα σε μία οντολογία ο όρος «Rock» θα μπορούσε να αναπαρασταθεί ως μία υποκλάση της «Music», η οποία θα μπορούσε περαιτέρω να συνδεθεί με σχετικές έννοιες και κλάσεις όπως «Instrument».Ενώ το λεξιλόγιο θα όριζε τι σημαίνει ο όρος «Rock» σε αυτό το πλαί-

σιο, διακρίνοντάς τον από το γεωλογικό «Rock» περιλαμβάνοντας σχολιασμούς σχετικά με τη χρήση του σε διαφορετικά μουσικά είδη ή στυλ.[8]

2.6.1 Βήματα δημιουργίας

Ο καθορισμός του πεδίου και του εύρους ενός λεξιλογίου αποτελεί ένα κρίσιμο βήμα για την δημιουργία του. Το πεδίο αναφέρεται στο αντικείμενο ή την περιοχή γνώσης που καλύπτει το λεξιλόγιο, ενώ το εύρος αναφέρεται στο πόσο εκτενώς καλύπτει αυτό το πεδίο. Μία σωστή προσέγγιση προϋποθέτει:

· **Ορισμός του Πεδίου** –Είναι σημαντικό να καθορίσουμε το πεδίο που θα καλύπτει το λεξιλόγιο. Αυτό μπορεί να είναι ένα ευρύ πεδίο, όπως η «ιατρική», ή ένα πιο συγκεκριμένο, όπως η «καρδιολογία». Η επιλογή του πεδίου θα εξαρτηθεί από τον σκοπό του λεξιλογίου και τη φύση του συστήματος σημασιολογικής αναζήτησης.

· **Είδος Χρήσης** –Για ποιο σκοπό θα χρεησιμοποιείτε το λεξιλόγιο και από ποίον. Για παράδειγμα, ένα σύστημα σημασιολογικής αναζήτησης για γενική αναζήτηση στον ιστό, μπορεί να χρειαστείτε ένα λεξιλόγιο με ευρεία έκταση και υψηλό εύρος που να καλύπτει μια μεγάλη ποικιλία θεμάτων. Αντίθετα, αν ένα σύστημα για ένα εξειδικευμένο πεδίο, όπως η ακαδημαϊκή έρευνα σε ένα συγκεκριμένο θέμα, μπορεί να χρειαστεί ένα λεξιλόγιο με πιο περιορισμένη έκταση αλλά μεγαλύτερο βάθος στο συγκεκριμένο πεδίο.

· **Αναγνώριση Κύριων Εννοιών και Όρων** – Εντοπισμός των κύριων εννοιών και όρων για το συγκεκριμένο πεδίο. Αυτό πρέπει να περιλαμβάνει όχι μόνο τους διαθέσιμους και γενικούς όρους, αλλά και εξειδικευμένη ορολογία που χρησιμοποιούν οι ειδικοί στον τομέα. Οι βασικοί όροι θα βοηθήσουν να διαμορφωθεί το εύρος του λεξιλογίου.

· **Ισορροπία Μεταξύ Πληρότητας και Διαχειρισιμότητας** – Θα πρέπει να υπάρχει ισορροπία μεταξύ πληρότητας και διαχειρισιμότητας. Ένα πιο ολοκληρωμένο λεξιλόγιο μπορεί να βελτιώσει την ακρίβεια και τη συνάφεια των αποτελεσμάτων της σημασιολογικής αναζήτησης. Ωστόσο, αν το λεξιλόγιο είναι υπερβολικά μεγάλο ή περίπλοκο, μπορεί να είναι δύσκολο να διατηρηθεί και να επιβραδύνει την διαδικασία. Ίσως χρειαστούν συμβιβασμοί ανάλογα με τους διαθέσιμους πόρους και τις ανάγκες του συστήματός.

· **Σχεδιασμός για Επέκταση** – Ακόμη και μετά τον καθορισμό της αρχικής έκτασης και του εύρους, το λεξιλόγιο θα πρέπει να είναι ευέλικτο και προσαρμόσιμο. Καθώς το πεδίο εξελίσσεται, θα χρειαστεί προσθήκη νέων όρων, αφαίρεση παρωχημένων και προσαρμογή των σχέσεων μεταξύ των όρων.

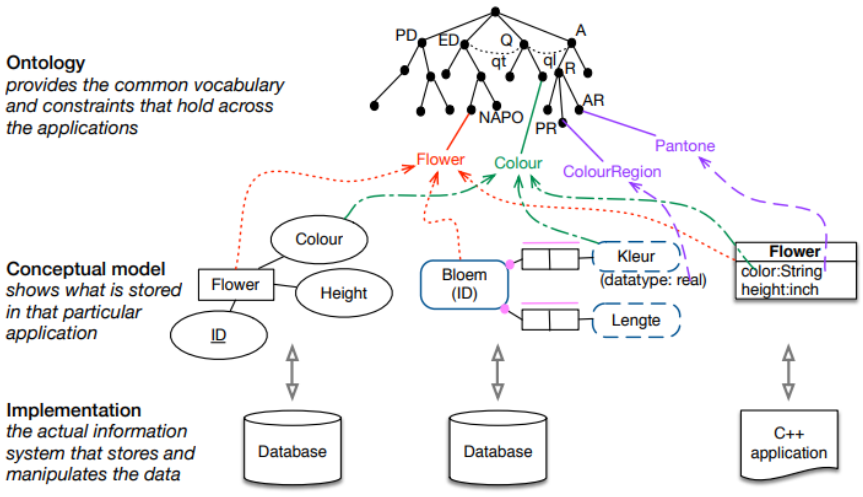
Συνοπτικά, ο καθορισμός της έκτασης και του εύρους είναι μια δυναμική διαδικασία που απαιτεί βαθιά κατανόηση του πεδίου, σαφή αντίληψη της χρήσης του και προθυμία για προσαρμογή και εξέλιξη.

2.6.2 Χρησιμότητα Οντολογιών

Ενοποίηση συστημάτων δεδομένων και πληροφοριών

Η οντολογική κατεύθυνση ενοποίησης δεδομένων αντιμετωπίζει την πρόκληση της συγχώνευσης πολλαπλών βάσεων δεδομένων ή συστημάτων που περιγράφουν τον ίδιο τομέα αλλά έχουν διαφορετική δομή. Αυτή η κατάσταση προκύπτει συχνά σε συγχωνεύσεις, συνεργασίες ή αναβαθμίσεις συστημάτων. Κάθε βάση δεδομένων μπορεί να έχει τη δική της δομή, ορολογία και μορφή, γεγονός που καθιστά την ενοποίηση δύσκολη.

Στο παρακάτω σχήμα διαφορετικά εννοιολογικά μοντέλα, όπως UML, EER και ORM αντιπροσωπεύουν την ίδια ιδέα με διαφορετικούς τρόπους, με το UML να αντιμετωπίζει το χρώμα ως χαρακτηριστικό του ”Flower”, ενώ σε ένα άλλο σύστημα το ”Kleur” (ολλανδικά για το ”colour”) μπορεί να αντιμετωπίζεται ως ξεχωριστή οντότητα. Παρά το γεγονός ότι αντιπροσωπεύουν την ίδια έννοια, οι διαφορετικές αναπαραστάσεις δημιουργούν ένα εννοιολογικό χάσμα. Οι οντολογίες διαδραματίζουν κρίσιμο ρόλο στη γεφύρωση αυτού του χάσματος, παρέχοντας ένα τυποποιημένο πλαίσιο για τον ορισμό και την ευθυγράμμιση αυτών των διαφορετικών εννοιών. Εγκαθιστούν ότι το ”Flower” σε διαφορετικά μοντέλα αναφέρεται στην ίδια υποκείμενη οντότητα και ότι το ”Colour” και το ”Kleur” είναι εννοιολογικά ισοδύναμα, ακόμη κι αν μοντελοποιούνται διαφορετικά. Οι οντολογίες επίσης διευκρινίζουν τις σχέσεις, όπως το χρώμα ενός λουλουδιού να εξαρτάται από την ύπαρξη του λουλουδιού, διασφαλίζοντας λογική συνέπεια.



Σχήμα 2.3: Σενάριο οντολογικής ενοποίησης εφαρμογών.

Αυτή η προσέγγιση διασφαλίζει ότι τα διαφορετικά συστήματα ευθυγραμμίζονται κάτω από μια κοινή κατανόηση όρων και σχέσεων, επιτρέποντας τη συνέπεια και τη διαλειτουργικότητα. Στο κάτω μέρος είναι οι διαφορετικές υλοποιήσεις, όπως σχεσιακές βάσεις δεδομένων και λογισμικό. Στο κέντρο τα εννοιολογικά μοντέλα δεδομένων προσαρμοσμένα στην εφαρμογή (ένα τμήμα ενός EER, ORM και UMLδιάγραμμα, αντίστοιχα).Και στην κορυφή η οντολογία που παρέχει ένα κοινό λεξιλόγιο για διαλειτουργικότητα μεταξύ των εφαρμογών.

Οι οντολογίες υποστηρίζουν επίσης τη λογική συμπερασματολογία, επιτρέποντας στα συστήματα να εξάγουν νέα γνώση από τις υπάρχουσες πληροφορίες. Ένα σύστημα υγείας που χρησιμοποιεί ιατρικές οντολογίες μπορεί να συμπεράνει ότι εάν ένας ασθενής έχει διαγνωστεί με διαβήτη, τότε είναι πιθανό να έχει υψηλά επίπεδα σακχάρου, ακόμη και αν αυτό δεν έχει καταγραφεί ρητά. Με αυτόν τον τρόπο, τα δεδομένα γίνονται πιο χρήσιμα χωρίς την ανάγκη ανθρώπινης παρέμβασης.
Ένας άλλος τομέας εφαρμογής των οντολογιών είναι η βελτίωση της αναζήτησης και της ανάκτησης πληροφοριών. Αντί για μια απλή αναζήτηση λέξεων-κλειδιών, οι σημασιολογικές μηχανές αναζήτησης κατανοούν το πλαίσιο των ερωτημάτων των χρηστών.
Οι οντολογίες και τα λεξιλόγια χρησιμοποιούνται επίσης ευρέως σε γραφήματα γνώσης, όπως το Google Knowledge Graph και το DBpedia. Αυτά τα συστήματα συνδέουν δεδομένα από διαφορετικές πηγές, επιτρέποντας την παροχή πιο ολοκληρωμένων και έξυπνων απαντήσεων.
Τέλος, οι οντολογίες βελτιώνουν την αυτοματοποίηση σε διάφορους τομείς, από το ηλεκτρονικό εμπόριο έως τις έξυπνες πόλεις.

2.6.3 Κοινές Οντολογίες και Λεξιλόγια

Υπάρχουν αρκετές καθιερωμένες οντολογίες και λεξιλόγια που έχουν υιοθετηθεί αναδεικνίοντας την ποικιλομορφία και την προσαρμοστικότητά τους. Αυτά τα πλαίσια έχουν γίνει πρότυπα στη βιομηχανία λόγω της εκτεταμένης κάλυψης κοινών τομέων και της συμβατότητάς τους με διάφορες τεχνολογίες. Ανάλογα με τις συγκεκριμένες ανάγκες ενός έργου, μπορεί να επιλεγεί η χρήση αυτών των καθιερωμένων πλαισίων ή ένας συνδυασμός. [9] [8]

Schema.org δημιουργήθηκε το 2011 από τις μεγαλύτερες μηχανές αναζήτησης, συμπεριλαμβανομένων των Google, Microsoft, Yahoo! και Yandex. Ο κύριος στόχος του είναι η βελτίωση της οργάνωσης και της αναζήτησης δεδομένων στο διαδίκτυο, μέσω της παροχής ενός κοινού λεξιλογίου για την περιγραφή ιστοσελίδων. Το Schema.org χρησιμοποιείται κυρίως στη σήμανση δεδομένων με RDFa, JSON-LD ή Microdata, και εφαρμόζεται σε επίπεδο RDF Schema (RDFS), χωρίς να υποστηρίζει πλήρως τις δυνατότητες του OWL. Χρησιμοποιείται σε τομείς όπως το ηλεκτρονικό εμπόριο, τα νέα, οι ταινίες, τα γεγονότα και πολλές άλλες κατηγορίες περιεχομένου, επιτρέποντας στις μηχανές αναζήτησης να εμφανίζουν πιο δομημένες πληρο-

φορίες, όπως τα πλούσια αποσπάσματα (rich snippets).

FOAF (Friend of a Friend) είναι ένα λεξιλόγιο που αναπτύχθηκε από τον Dan Brickley και τον Libby Miller στις αρχές της δεκαετίας του 2000. Χρησιμοποιείται για την περιγραφή των κοινωνικών σχέσεων και των προσωπικών προφίλ στον Σημασιολογικό Ιστό. Το FOAF βασίζεται σε RDF και επιτρέπει στους χρήστες να περιγράφουν πληροφορίες όπως το όνομα, το email, οι φίλοι και οι σχέσεις μεταξύ ατόμων και οργανισμών. Χρησιμοποιείται κυρίως σε κοινωνικά δίκτυα, εφαρμογές αναγνώρισης ταυτότητας και διαμοιρασμού δεδομένων, επιτρέποντας τη διασύνδεση των χρηστών μεταξύ διαφορετικών πλατφορμών με τρόπο που δεν εξαρτάται από κεντρικούς διακομιστές.

Dublin Core είναι ένα λεξιλόγιο μεταδεδομένων που αναπτύχθηκε από την Dublin Core Metadata Initiative (DCMI) τη δεκαετία του 1990. Δημιουργήθηκε με σκοπό την παροχή ενός απλού και ευέλικτου τρόπου περιγραφής πηγών πληροφορίας, όπως βιβλία, άρθρα, εικόνες και ιστότοποι. Το Dublin Core υποστηρίζει τόσο βασικά (simple) όσο και εμπλουτισμένα (qualified) μεταδεδομένα και είναι βασισμένο σε RDF και XML. Χρησιμοποιείται κυρίως σε βιβλιοθήκες, αρχεία, μουσεία και ακαδημαϊκές βάσεις δεδομένων για τη βελτίωση της αναζήτησης και της οργάνωσης πληροφοριών.

GeoNames Ontology αναπτύχθηκε για τη διασύνδεση γεωγραφικών δεδομένων και την περιγραφή τοποθεσιών σε παγκόσμιο επίπεδο. Η βάση δεδομένων GeoNames περιέχει εκατομμύρια τοποθεσίες και συνδέεται με πληροφορίες όπως γεωγραφικά όρια, υψόμετρο, πληθυσμός και διοικητική διαίρεση. Η οντολογία του GeoNames βασίζεται σε RDF και OWL, επιτρέποντας τη σημασιολογική σύνδεση τοποθεσιών με άλλες πηγές δεδομένων, όπως η DBpedia και το Linked Open Data cloud. Χρησιμοποιείται ευρέως σε γεωγραφικά συστήματα πληροφοριών (GIS), τουριστικές εφαρμογές και χάρτες δεδομένων, διευκολύνοντας την αναζήτηση και την ενοποίηση γεωγραφικών πληροφοριών.

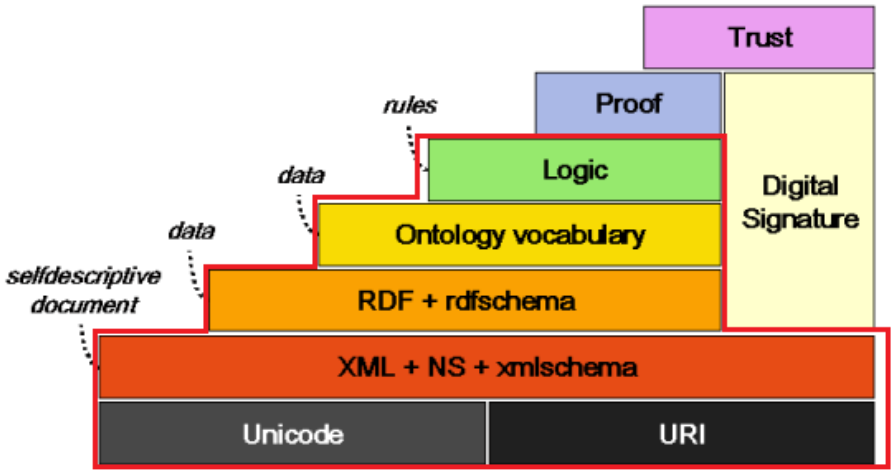
SNOMED CT (Systematized Nomenclature of Medicine - Clinical Terms) είναι μια από τις πιο προηγμένες ιατρικές οντολογίες, η οποία αναπτύχθηκε από τη SNOMED International. Η αρχική της έκδοση δημιουργήθηκε τη δεκαετία του 1970, αλλά εξελίχθηκε σημαντικά και το 2007 ενώθηκε με το Clinical Terms (CT), δημιουργώντας το SNOMED CT. Αυτή η οντολογία περιέχει χιλιάδες όρους που σχετίζονται με την ιατρική, επιτρέποντας τη δομημένη αναπαράσταση ασθενειών, συμπτωμάτων, διαδικασιών και άλλων κλινικών εννοιών. Χρησιμοποιείται κυρίως στα ηλεκτρονικά ιατρικά αρχεία, την υγειονομική περίθαλψη και τη βιοϊατρική έρευνα, επιτρέποντας την ακριβέστερη καταγραφή και ανταλλαγή δεδομένων μεταξύ διαφορετικών συστημάτων.

Κεφάλαιο 3

Βασικές Τεχνολογίες

Εισαγωγή

Η στοίβα του Σημασιολογικού Ιστού (Semantic Web Stack), γνωστή και ως Layer Cake, αντιπροσωπεύει την αρχιτεκτονική του Σημασιολογικού Ιστού. Αποτελείται από μια ιεραρχία γλωσσών, όπου κάθε κατώτερο επίπεδο παρέχει βασικές δυνατότητες στο επίπεδο που βρίσκεται από πάνω. Κάθε επίπεδο συνδέεται με συγκεκριμένα πρότυπα και προδιαγραφές, επιτρέποντας τη διαλειτουργικότητα και την ανταλλαγή δομημένων δεδομένων. Σε αυτό το κεφάλαιο θα αναλύσουμε τα βασικά στρώματα που είναι πλήρως τυποποιημένα και ευρέως αποδεκτά για την ανάπτυξη οντολογιών που φαίνονται στο κόκκινο περίγραμμα.Τα ανώτερα επίπεδα δεν έχουν ακόμη καθοριστεί με σαφήνεια.



Σχήμα 3.1: Τα επίπεδα του σημασιολογικού ιστού.

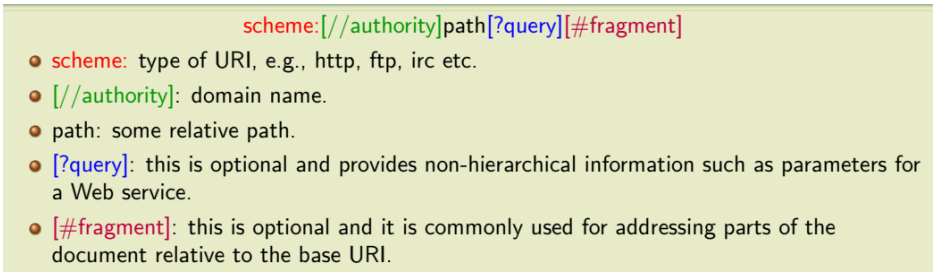
Για τη δημιουργία του μοντέλου, είναι σημαντικό να κατανοήσουμε τόσο τη λειτουργία και τις

δυνατότητες που προσφέρει κάθε επίπεδο όσο και τον τρόπο με τον οποίο συνδέονται μεταξύ τους. Το πως τα ανώτερα επίπεδα βασίζονται και αξιοποιούν τα κατώτερα, δημιουργώντας μια ιεραρχική δομή που επιτρέπει την οργάνωση, την επέκταση και τη συσχέτιση των δεδομένων με συνέπεια και ακρίβεια.
Το πρώτο επίπεδο παρέχει τις τεχνολογίες για τον μηχανισμό μοναδικής αναγνώρισης πόρων. Στο δεύτερο επίπεδο περιέχονται οι χώροι ονομάτων (namespaces) για την ονομασία ομάδας στοιχείων, XML (markup language) για την αναπαράσταση εγγράφων με χρήση tags και XML schema (metalanguage) για τον ορισμό της δομής των εγγράφων αυτών. Τρίτο επίπεδο RDF και RDFS για την περιγραφή μεταδεδομένων. Στο επίπεδο τέσσερα βρίσκονται τα οντολογικά λεξιλόγια που επεκτείνουν τις δυνατότητες του Rdfs με νέες έννοιες και προσθέτοντας περισσότερη σημασιολογία. Στο πέμπτο επίπεδο βρίσκονται γλώσσες που βασίζονται στη λογική για εξαγωγή συμπερασμάτων (SWRL,RIF,SPIN). Το έκτο αποτελεί το επίπεδο απόδειξης όπου ελέγχει την εγκυρότητα των συμπερασμάτων. Ενώ το έβδομο ορίζει τους μηχανισμούς εμπιστοσύνης για την αξιοπιστία των δεδομένων.

3.1 IRI,URI,URL

URI(Uniform Resource Identifier) είναι απλά μια συμβολοσειρά ASCII που χρησιμοποιείται για την αναγνώριση πόρων. Ένας πόρος δεν είναι απαραίτητα προσβάσιμος μέσω του Διαδικτύου· για παράδειγμα, ανθρώπινα όντα, εταιρείες, κτίρια, έργα τέχνης, ποτάμια και βιβλία σε μια βιβλιοθήκη μπορούν όλα να θεωρηθούν πόροι. Αφηρημένες έννοιες μπορούν επίσης να είναι πόροι. Άλλοι όροι που χρησιμοποιούνται για τους πόρους είναι η οντότητα και το αντικείμενο (thing).
Μια οντότητα μπορεί να έχει ένα αναγνωριστικό κωδικό που έχει καθιερωθεί σε μια βάση δεδομένων αρχών, όπως το Library of Congress/NACO Authority File μέσω της υπηρεσίας LC Linked Data Service (http://id.loc.gov), ή από μια υπηρεσία που δημιουργεί αναγνωριστικά, όπως το Wikidata (http://wikidata.org) ή το BBC Things (http://www.bbc.co.uk/things/).
Ένα αναγνωριστικό που κατασκευάζεται με ένα πρωτόκολλο υπηρεσίας ιστού ως πρόθεμα, π.χ. http://, αναφέρεται ως HTTP URI. Στο περιβάλλον του Resource Description Framework (RDF), ένα HTTP URI είναι ένα ανακτήσιμο (dereferenceable) URI που διευκολύνει τις λειτουργίες από μηχανή σε μηχανή. Τα URI μπορούν να ταξινομηθούν είτε ως URL είτε ως URN. Εκτός από την αναγνώριση ενός πόρου τα URLs παρέχουν και έναν τρόπο εντοπισμού του, περιγράφοντας τον κύριο μηχανισμό πρόσβασης, π.χ http://, ftp://, mailto:, κ.λπ. Αντίθετα, τα URNs αναγνωρίζουν μοναδικά έναν πόρο, αλλά δεν καθορίζουν απαραίτητα την τοποθεσία του ή τον τρόπο πρόσβασης σε αυτόν.
Κάθε URI ξεκινά με ένα όνομα σχήματος, το οποίο αναφέρεται σε μια προδιαγραφή για την ανάθεση αναγνωριστικών εντός αυτού του σχήματος. Υπάρχουν δεκάδες σχήματα, αλλά τα πιο

κοινά στις εφαρμογές είναι τα http, https, ftp και mailto. Το όνομα του σχήματος ακολουθείται πάντα από άνω και κάτω τελεία (:) ακολουθεί το όνομα του τομέα (domain) και η διοδρομή (path) .



Σχήμα 3.2: Μορφή URI.

Παραδείγματα URI:
urn:isbn:0-679-73669-7
https://doi.org/10.1037/arc0000014

IRI(Internationalized Resource Identifier)

Το διεθνοποιημένο αναγνωριστικό πόρων (IRI) είναι ένα τυποποιημένο αναγνωριστικό που χρησιμοποιείται για την μοναδική ονομασία πόρων στο διαδίκτυο ή σε συγκεκριμένα συστήματα. Είναι μια εκτεταμένη έκδοση ενός URI (Uniform Resource Identifier) που υποστηρίζει ένα ευρύτερο φάσμα χαρακτήρων, συμπεριλαμβανομένων χαρακτήρων Unicode, όπως αυτοί που χρησιμοποιούνται σε μη λατινικά σενάρια (π.χ. ελληνικά, αραβικά ή κινέζικα). Τα IRI καθιστούν την αναγνώριση πόρων πιο περιεκτική και παγκόσμια εφαρμόσιμη, επιτρέποντας γλώσσες και σύμβολα πέρα από το σύνολο χαρακτήρων ASCII.
Στο RDF (Πλαίσιο Περιγραφής Πόρων), τα IRI είναι απαραίτητα επειδή παρέχουν έναν τρόπο για τον μοναδικό εντοπισμό πόρων (όπως οντότητες, ιδιότητες ή κλάσεις) σε ένα κατανεμημένο και παγκόσμιο περιβάλλον. Το RDF είναι ένα πλαίσιο για την αναπαράσταση πληροφοριών σχετικά με πόρους σε μια δομή γραφήματος και κάθε κόμβος ή ακμή σε αυτό το γράφημα πρέπει να προσδιορίζεται μοναδικά για να διασφαλίζεται η διαλειτουργικότητα. Χωρίς IRI, θα ήταν δύσκολο να αποσαφηνιστούν οι πόροι σε διαφορετικά συστήματα, οδηγώντας σε ασυνέπειες και συγκρούσεις. Τα IRI επιτρέπουν στο RDF να υποστηρίζει πολύγλωσσα και διεθνοποιημένα δεδομένα, κάτι που είναι κρίσιμο σε έναν κόσμο όπου το περιεχόμενο Ιστού δημιουργείται και καταναλώνεται σε διάφορες γλώσσες. Χρησιμοποιώντας IRI, το RDF μπορεί να ενσωματώσει και να αναπαραστήσει αποτελεσματικά δεδομένα από διαφορετικές πηγές, επιτρέποντας σε εφαρμογές όπως ο σημασιολογικός ιστός, τα συνδεδεμένα δεδομένα και τα γραφήματα γνώσης να λειτουργούν παγκοσμίως. Στην πράξη, τα IRI σε RDF συχνά γράφονται ως μεγάλα URI ή συμπαγείς μορφές που ονομάζονται CURIE (Compact URIs), που χρησιμο-

ποιούν ένα πρόθεμα για συντομία.
Η κύρια διαφορά μεταξύ ενός IRI και ενός URL έγκειται στο σύνολο χαρακτήρων και τον σκοπό τους. Ενώ ένα URL είναι ένα υποσύνολο URI που καθορίζει τη θέση ενός πόρου και τον τρόπο πρόσβασης σε αυτόν.Μια διεύθυνση URL μπορεί να θεωρηθεί ένας συγκεκριμένος τύπος IRI που εστιάζει στην ανάκτηση πόρων μέσω πρωτοκόλλου (όπως HTTP ή HTTPS). Ωστόσο, δεν είναι όλα τα IRI URL επειδή ορισμένα IRI χρησιμοποιούνται για αναγνώριση χωρίς να είναι προσπελασιμα.

URLs ⊆ URIs ⊆ IRIs

3.2 RDF(Resource Description Framework)

Στις μέρες μας, όλο και περισσότερες συσκευές παράγουν δεδομένα αυτόματα, ενώ είναι σχετικά εύκολο να αναπτυχθούν εφαρμογές σε διάφορους τομείς που βασίζονται σε βάσεις δεδομένων και εκθέτουν δεδομένα στον Ιστό. Η ποσότητα και η ποικιλία των δεδομένων που παράγονται ξεπερνούν κατά πολύ την ικανότητά μας να τα καταναλώσουμε.
Ο όρος *big data* χρησιμοποιείται για να περιγράψει δεδομένα τόσο μεγάλα και πολύπλοκα που οι παραδοσιακές εφαρμογές επεξεργασίας δεδομένων δεν μπορούν να τα διαχειριστούν. Τρεις λέξεις που μπορούν να χαρακτηρίσουν τον όρο αυτο είναι *όγκος*, *ταχύτητα* και *ποικιλία*. Παρόλο που ο όγκος και η ταχύτητα είναι τα πιο εμφανή χαρακτηριστικά, η ποικιλία αποτελεί σημαντικό ζήτημα, καθώς εμποδίζει την ενοποίηση των δεδομένων και δημιουργεί σοβαρά προβλήματα διαλειτουργικότητας.
Το RDF προτάθηκε ως ένα μοντέλο δεδομένων βασισμένο σε γράφους και ενσωματώθηκε στο όραμα του Σημασιολογικού Ιστού. Η αξιοποίηση των URI ως παγκόσμιων ταυτοποιητών προσέφερε μια λύση στο πρόβλημα της ενοποίησης δεδομένων, καθώς τα RDF σύνολα δεδομένων που παράγονται από διαφορετικές πηγές μπορούν να ενσωματωθούν απρόσκοπτα με άλλα δεδομένα. Η ενοποίηση δεδομένων με RDF είναι ταχύτερη και πιο ανθεκτική από τις παραδοσιακές λύσεις, ειδικά όταν προκύπτουν αλλαγές στα σχήματα δεδομένων.
Το RDF είναι επίσης βασικό στοιχείο των *συνδεδεμένων δεδομένων (linked data)*. Τα συνδεδεμένα δεδομένα προτάθηκαν ως ένα σύνολο βέλτιστων πρακτικών για τη δημοσίευση δεδομένων στον Ιστό. Ο όρος εισήχθη από τον Tim Berners-Lee και βασίζεται σε τέσσερις βασικές αρχές, μία εκ των οποίων αναφέρεται στο RDF ως ένα από τα πρότυπα που παρέχουν χρήσιμες πληροφορίες. Ο στόχος είναι η πληροφορία να μην είναι χρήσιμη μόνο για ανθρώπους που περιηγούνται μέσω προγραμμάτων περιήγησης (όπου το HTML θα ήταν αρκετό), αλλά και για άλλους παράγοντες που μπορούν να επεξεργαστούν αυτόματα τα δεδομένα.

3.2.1 Πλεονεκτήματα

Αποσαφήνιση (Disambiguation)
Η χρήση των IRI για την αναγνώριση ιδιοτήτων (predicates) και τη διατύπωση δηλώσεων σχετικά με πόρους επιτρέπει στον χρήστη να αναγνωρίζει παγκοσμίως την ιδιότητα που αποδίδεται, καθώς και τους πόρους που εμπλέκονται στη δήλωση. Αυτές οι παγκόσμιες ιδιότητες μπορούν να αναγνωριστούν από αυτοματοποιημένους πράκτορες, οι οποίοι κατανοούν τα δεδομένα με έναν μη-αμφίσημο τρόπο.

RDF ως γλώσσα ολοκλήρωσης (integration language)
Το RDF είναι συνθετικό από τη φύση του, καθώς δύο RDF γράφοι που προέρχονται από ανεξάρτητες πηγές μπορούν να συγχωνευθούν αυτόματα σε έναν μεγαλύτερο γράφο. Αυτή η ιδιότητα διευκολύνει την ενοποίηση δεδομένων από ετερογενείς πηγές.
Ένα από τα μεγαλύτερα προβλήματα στη σύγχρονη επιστήμη των υπολογιστών είναι η διαλειτουργικότητα μεταξύ εφαρμογών που επεξεργάζονται δεδομένα από διαφορετικές πηγές. Το RDF συμβάλλει στην επίλυση αυτού του προβλήματος, καθώς τα RDF δεδομένα μπορούν να ενοποιηθούν αυτόματα, ακόμη και αν έχουν παραχθεί από διαφορετικούς φορείς.

RDF ως κοινή γλώσσα για τον Σημασιολογικό Ιστό και τα Συνδεδεμένα Δεδομένα
Η απλότητα και η γενικότητα του μοντέλου δεδομένων του RDF το καθιστούν κατάλληλο για τη μοντελοποίηση κάθε είδους δεδομένων, διευκολύνοντας την ενοποίηση με άλλα δεδομένα. Το RDF αποτελεί τον πυρήνα του σημασιολογικού ιστού (Semantic Web Stack) και αναφέρεται στις αρχές των συνδεδεμένων δεδομένων (Linked Data Principles), καθώς και στο πενταεπίπεδο μοντέλο ποιότητας δεδομένων.

Αποθήκευση RDF δεδομένων και SPARQL
Το 2008, το SPARQL προτάθηκε ως γλώσσα ερωτημάτων για το RDF και έτυχε τεράστιας αποδοχής από την κοινότητα του RDF. Η δυνατότητα εκτέλεσης ερωτημάτων οδήγησε στην ανάπτυξη πολλών νέων εφαρμογών, βάσεων δεδομένων και βιβλιοθηκών.
Οι αποθήκες RDF (RDF data stores) άρχισαν να γίνονται δημοφιλείς, ενώ ορισμένες εταιρείες υιοθέτησαν το RDF για την εσωτερική αναπαράσταση των δεδομένων τους. Σε κάποιες περιπτώσεις, το RDF επιλέχθηκε αποκλειστικά για πρακτικούς λόγους, ανεξάρτητα από τη σχέση του με τον σημασιολογικό ιστό.Η αποθήκευση RDF και η χρήση του SPARQL προσφέρουν ένα ευέλικτο μοντέλο που μπορεί να προσαρμοστεί εύκολα σε αλλαγές του μοντέλου δεδομένων.

Επεκτασιμότητα (Extensibility)
Όταν αναπτύσσεται μια εφαρμογή, είναι σημαντικό η αποθήκευση δεδομένων να έχει δυνατότητα επέκτασης ώστε να προσαρμόζεται σε νέες ανάγκες. Το γραφικό μοντέλο του RDF

επιτρέπει την εύκολη προσθήκη νέων δηλώσεων σε οποιονδήποτε γράφο.

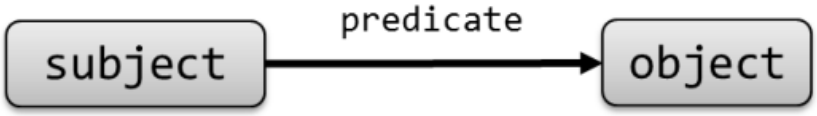
Ευελιξία (Flexibility)
Σε αντίθεση με τις σχεσιακές βάσεις δεδομένων, όπου οι αλλαγές στη δομή μπορεί να είναι δύσκολες, το RDF ενσωματώνει την ευελιξία, επιτρέποντας τροποποιήσεις απλώς με την ενημέρωση των τριπλετών (triples).

3.2.2 Data model

Το RDF (Resource Description Framework) είναι ένα πρότυπο που έχει αναπτυχθεί από το W3C για την αναπαράσταση πληροφοριών και δεδομένων στο Διαδίκτυο. Επιτρέποντας τη δημιουργία δομημένων συνόλων που μπορούν να κατανοηθούν και να επεξεργαστούν από μηχανές. Η θεμελιώδης αρχή του RDF είναι η αναπαράσταση δεδομένων μέσω τριάδων (triple), οι οποίες αποτελούνται από τα:

- Υποκείμενο (Subject)
- Κατηγόρημα (Predicate)
- Αντικείμενο (Object)

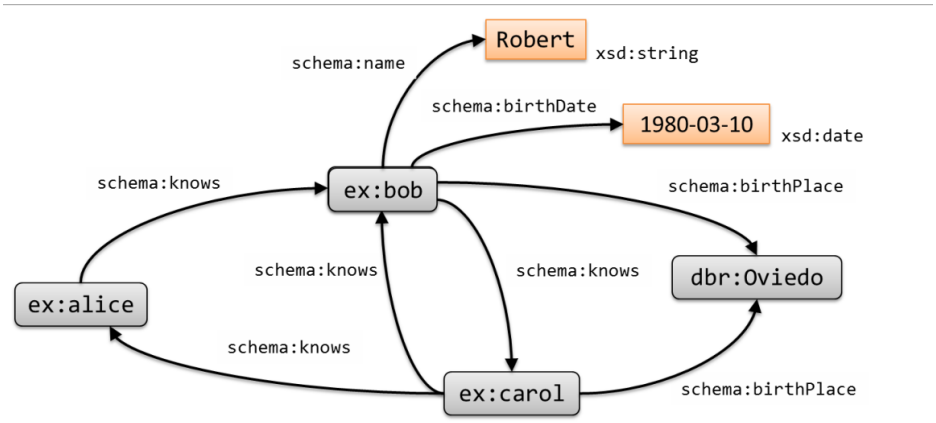
Το υποκείμενο αναπαριστά τον πόρο που περιγράφεται, το κατηγόρημα δηλώνει τη σχέση ή την ιδιότητα που περιγράφεται, και το αντικείμενο είναι η τιμή ή ο άλλος πόρος που συνδέεται. Μια τριάδα RDF που δηλώνεται σημαίνει ότι κάποια σχέση, που υποδεικνύεται από το



Σχήμα 3.3: Παράδειγμα RDF τριάδας.

κατηγόρημα, ισχύει μεταξύ του υποκειμένου και του αντικειμένου. Αυτό είναι γνωστό ως δήλωση RDF. Το κατηγόρημα είναι ένα IRI που δηλώνει μια ιδιότητα. Μια δήλωση rdf μπορεί να θεωρηθεί ως μια δυαδική σχέση που προσδιορίζεται από την ιδιότητα μεταξύ του υποκειμένου και του αντικειμένου. Στις δηλώσεις αυτες ή τριάδες το Υποκείμενο και το Αντικείμενο ονομάζονται κόμβοι.[10]
Μπορούν να υπάρχουν τρία είδη κόμβων: μοναδικό αναγνωριστικό(IRI), τυποποιημένα λεκτικά(literals) και κενά nodes.

Ένα IRI αναφέρεται σε έναν πόρο και μπορεί να είναι οτιδήποτε. Τα IRI μπορούν να εμφανίζονται ως υποκείμενα, κατηγορήματα και αντικείμενα. Ένα literal δηλώνει πόρους που έχουν μια συσχετισμένη τιμή, για παράδειγμα, μια ακέραια ή μία συμβολοσειρά. Τα literals μπορούν να εμφανίζονται μόνο ως αντικείμενα στις τριάδες. Περιέχουν μια λεκτική μορφή και ένα iri τύπου δεδομένων που αναπαρίστανται ως ”lexicalForm”datatype στο turtle. Για παράδειγμα: ”23”xsd:integer αναπαριστά έναν ακέραιο με τιμή 23 και ”1980-03-01”xsd:date αναπαριστά την 1η Μαρτίου 1980. Όλα τα literals στο rdf έχουν έναν συσχετισμένο τύπο δεδομένων. Στην περίπτωση των συμβολοσειρών χωρίς δηλωμένο τύπο δεδομένων, θεωρείται από προεπιλογή ο τύπος xsd:string. Τα κενά nodes είναι τοπικοί αναγνωριστές που δεν ταυτοποιούν συγκεκριμένους πόρους. Τα κενά nodes μπορούν να χρησιμοποιηθούν ως υποκείμενα ή αντικείμενα σε τριπλέτες. Δηλώνουν ότι κάτι με τη δεδομένη σχέση υπάρχει, χωρίς να το ονομάζουν ρητά. Ένα γράφημα RDf είναι ένα σύνολο από δηλώσεις . Οι ακμές των γραφημάτων μπορούν να είναι μόνο IRI. Αυτό είναι ένα σημαντικό χαρακτηριστικό του rdf που επιτρέπει την παγκόσμια ταυτοποίηση των κατηγορημάτων που δηλώνονται. Τα υποκείμενα μπορούν να είναι μόνο IRIs ή κενά nodes. Ενώ τα αντικείμενα μπορούν να είναι IRIs, κενά nodes ή literals.



Σχήμα 3.4: Παράδειγμα RDF graph.

Ένα σημαντικό χαρακτηριστικό των γραφημάτων RDF είναι ότι δύο ανεξάρτητα γραφήματα RDF μπορούν να συγχωνευθούν αυτόματα για να ληφθεί ένα μεγαλύτερο γράφημα RDF που σχηματίζεται από την ένωση των δηλώσεων. Δεδομένης της παγκόσμιας φύσης των IRI, οι κόμβοι με το ίδιο IRI ενοποιούνται αυτόματα. Η χρήση κοινών IRI καθιστά την ισχυρή δήλωση ότι οι οντότητες και οι σχέσεις σε ένα γράφημα έχουν την ίδια πρόθεση όπως και στα άλλα γραφήματα χρησιμοποιώντας τα ίδια αναγνωριστικά. [11]

3.2.3 Namespaces

Namespaces(χώροι ονομάτων) είναι ένας μηχανισμός για την οργάνωση και την αποσαφήνιση των IRI που χρησιμοποιούνται στα δεδομένα RDF. Ένας χώρος ονομάτων είναι ουσιαστικά μια συντομογραφία για ένα μεγαλύτερο IRI που αντιπροσωπεύει ένα συγκεκριμένο λεξιλόγιο ή σχήμα. Επιτρέπει στους προγραμματιστές να επαναχρησιμοποιούν όρους που ορίζονται σε εξωτερικά λεξιλόγια, αποφεύγοντας ταυτόχρονα τις συγκρούσεις ονομάτων όταν συνδυάζουν δεδομένα από διαφορετικές πηγές. Συσχετίζοντας ένα σύντομο πρόθεμα με ένα βασικό IRI, οι χώροι ονομάτων κάνουν τα δεδομένα RDF πιο ευανάγνωστα και διαχειρίσιμα. Για παράδειγμα, το πρόθεμα foaf: χρησιμοποιείται συνήθως για να αναφέρεται σε όρους από την οντολογία Friend of a Friend (FOAF), όπως foaf:name. Οι χώροι ονομάτων είναι ιδιαίτερα σημαντικοί για την προώθηση της διαλειτουργικότητας και της επαναχρησιμοποίησης στο RDF, καθώς επιτρέπουν στους δημιουργούς δεδομένων να βασίζονται σε υπάρχουσες οντολογίες και λεξιλόγια αντί να ορίζουν νέους όρους από την αρχή. Διασφαλίζουν επίσης ότι κάθε όρος παραμένει μοναδικός στο ευρύτερο πλαίσιο των συνδεδεμένων δεδομένων.

Παράδειγμα χρήσης Iri και namespaces:
@prefix ex: <http://example.com/> .
ex:Person1 ex:hasName “giannis” .
Ορίζουμε ως namespace το πρόθεμα ex: για το IRI http://example.com το οποίο μας επιτρέπει την εύκολη επαναχρησιμοποίηση του θέτοντάς το ως βάση και άρα το IRI που προσδιορίζει μοναδικά το person1 είναι το http://example.com/Person1.

3.2.4 Μορφές Σύνταξης

Το μοντέλο δεδομένων RDF χρειάζεται μορφοποίηση για να μπορεί να ανταλλάσσεται και να αποθηκεύεται. Αν και αρχικά βασίστηκε στο XML, οι σύγχρονες υλοποιήσεις δεν το απαιτούν πλέον. Υπάρχουν πολλαπλές μορφές σειριοποίησης πέρα από το RDF/XML, όπως Turtle, N3 και TriX, RDFa οι οποίες αποθηκεύουν RDF ως αυτόνομα έγγραφα με κατάλληλο τύπο περιεχομένου για την αναγνώρισή τους από τους πελάτες.

RDF/XML: Η αρχική ιδέα για την αναπαράσταση του RDF ήταν η χρήση XML μορφοποίησης (http://www.w3.org/TR/rdf-syntax-grammar/). Σε γενικές γραμμές, κάθε πόρος μπορεί να οριστεί ως ένα στοιχείο XML τύπου rdf:Description, με ένα χαρακτηριστικό rdf:about που δηλώνει το URI του. Πολλαπλά χαρακτηριστικά που σχετίζονται με ένα συγκεκριμένο υποκείμενο δίνονται ως θυγατρικά στοιχεία του αντίστοιχου XML στοιχείου. Για να γίνει αναφορά σε ένα συγκεκριμένο URI, θα πρέπει να χρησιμοποιείται το χαρακτηριστικό rdf:resource.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22 -rdf-syntax-ns#"
         xmlns:blog="http://example.com/Blog#"
         xmlns:ex="http://example.com/terms#">
  <rdf:Description rdf:about="htt p://example.com/Blog#JD">
      <rdf:type rdf:resource="http://example.com/Blog#User"/>
      <blog:hasGender rdf:resource="http://example.com/terms#Male"/>
      <ex:firstName>Joe</ex:firstName>
      <ex:lastName>Doe</ex:lastName>
  </rdf:Description>
</rdf:RDF>
```

Σχήμα 3.5: RDF/XML .

RDFa: Επιτρέπει την ενσωμάτωση RDF δεδομένων μέσα σε HTML ή XHTML έγγραφα, χρησιμοποιώντας ειδικά χαρακτηριστικά HTML.

```
<div vocab="https://schema.org/" typeof="Person">
    <p>
        <span property="name">Amy Rogers</span>
        <span property="jobTitle">Founder/CEO</span>
        <span property="telephone">Phone: (540) 961-4469</span>
    </p>
    <p>E-mail: <span property="email">arogers@digitalar.com</span> </p>
    <p> Links: <span property="url">Amy's Homepage</span> </p>
</div>
```

Σχήμα 3.6: RDFa .

Turtle Γίνεται ολοένα και πιο δημοφιλής γλώσσα και αποτελεί πλέον σύσταση του W3C. Ως κατηγόρημα, το a ισοδυναμεί με το πλήρες URI που αντιστοιχεί στο rdf:type. Οι κενοί κόμβοι RDF μπορούν είτε να εκφραστούν ρητά ως _:id, όπου id είναι το αναγνωριστικό του κενού κόμβου, είτε ανώνυμα τοποθετώντας όλες τις τριάδες που είναι υποκείμενα μεταξύ αγκυλών. Επίσης παρέχει μια δομή συλλογής για λίστες ως ακολουθία στοιχείων που χωρίζονται με κενά και περικλείονται από παρενθέσεις.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix cat: <http://example.com/Cat#> .
@prefix blog: <http://example.com/Blog#> .
@prefix ex: <http://example.com/terms#> .
…
blog:MS     a                 blog:User ;
            ex:firstName        "Mary" ;
            ex:lastName         "Smith" ;
            blog:hasGender    ex:Female ;
            blog:isFollowing  ( blog:JD  _:blogger4 ) .
_:blogger4  a                 blog:User .
…
```

Σχήμα 3.7: Turtle .

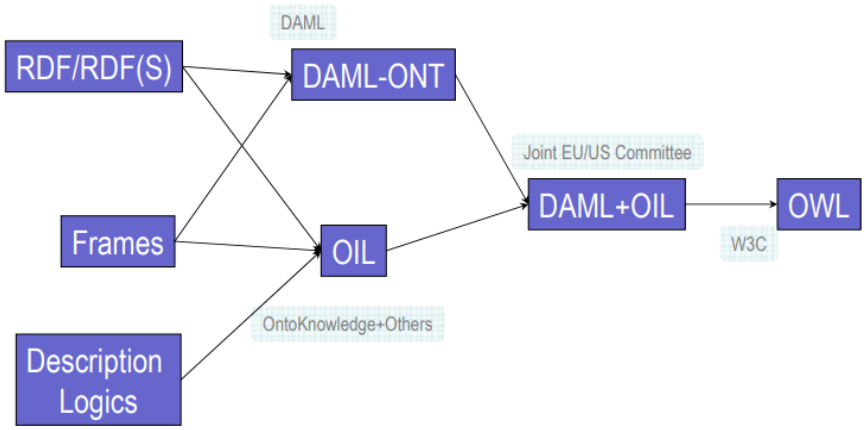
3.3 RDFS

Το RDFS (RDF Schema) είναι μια επέκταση του RDF που παρέχει μηχανισμούς για την περιγραφή της σημασιολογίας των δεδομένων. Ενώ το RDF επικεντρώνεται στην αναπαράσταση δεδομένων, το RDFS επεκτείνει αυτή τη δυνατότητα, προσφέροντας εργαλεία για τη δημιουργία δομημένων μοντέλων δεδομένων. Μέσω του RDFS, οι χρήστες μπορούν να ορίσουν κλάσεις, ιδιότητες, ιεραρχίες και περιορισμούς που καθορίζουν τη σχέση μεταξύ διαφορετικών στοιχείων ενός συνόλου δεδομένων. Ένα από τα βασικά χαρακτηριστικά του RDFS είναι η δυνατότητα ορισμού κλάσεων για την κατηγοριοποίηση πόρων. Οι κλάσεις επιτρέπουν την ομαδοποίηση παρόμοιων αντικειμένων και τη δημιουργία ιεραρχιών μέσω της ιδιότητας rdfs:subClassOf. Παρομοίως, οι ιδιότητες μπορούν να οργανωθούν σε ιεραρχίες με τη χρήση της rdfs:subPropertyOf, επιτρέποντας την αναπαράσταση πιο εξειδικευμένων σχέσεων. Το RDFS υποστηρίζει επίσης τον καθορισμό του πεδίου (rdfs:domain) και του εύρους (rdfs:range) των ιδιοτήτων. Το πεδίο μιας ιδιότητας ορίζει την κλάση των αντικειμένων που μπορούν να λειτουργήσουν ως υποκείμενα της ιδιότητας, ενώ το εύρος καθορίζει την κλάση των αντικειμένων που μπορούν να λειτουργήσουν ως αντικείμενα. Αυτές οι δυνατότητες συμβάλλουν στην εξασφάλιση της συνέπειας και της ακρίβειας των δεδομένων. Το RDFS χρησιμοποιείται σε εφαρμογές όπου απαιτείται η μοντελοποίηση και η κατηγοριοποίηση σύνθετων δεδομένων. Μπορεί να λειτουργήσει αυτόνομα ή σε συνδυασμό με άλλες τεχνολογίες, όπως η OWL, για την ανάπτυξη σύνθετων οντολογιών που απαιτούν μεγαλύτερη εκφραστικότητα. Το RDFS, αν και λιγότερο εκφραστικό από την OWL, παρέχει μια ισχυρή βάση για την περιγραφή και τη διαχείριση δεδομένων, ενισχύοντας τη διαλειτουργικότητα και την επαναχρησιμοποίηση τους σε διαφορετικά περιβάλλοντα.[12]

3.4 OWL

OWL (Web Ontology Language) είναι μια γλώσσα που σχεδιάστηκε για να περιγράφει πολύπλοκες έννοιες, σχέσεις και περιορισμούς σε έναν τομέα γνώσης συμβατή με τα υπάρχοντα πρότυπα του Ιστού. Είναι τυποποιημένη από το W3C (World Wide Web Consortium) και αποτελεί σημαντικό εργαλείο για τη δημιουργία οντολογιών. Σημείο εκκίνησης αποτέλεσε η γλώσσα DAML(Darpa Agent Markup Language) + OIL (Ontology Inference Layer), η οποία προέκυψε από τη συγχώνευση της αμερικανικής πρότασης DAML-ONT και της ευρωπαϊκής γλώσσας OIL, με στόχο να γίνει ευρέως αποδεκτή γλώσσα οντολογιών. [13]

Η OWL είναι μια γλώσσα που βασίζεται σε υπολογιστική λογική, έτσι ώστε η γνώση που εκφράζεται να μπορεί να αξιοποιηθεί από προγράμματα υπολογιστών. Βασίζεται στο RDF και στο RDFS και χρησιμοποιεί παρόμοια σύνταξη χρησιμοποιώντας XML. Επεκτείνοντας τις δυνα-



Σχήμα 3.8: Εισαγωγή στην Owl.

τότητές τους προσφέροντας μεγαλύτερη εκφραστικότητα. Στην OWL, η τυπική σημασιολογία βασίζεται στη Λογική Περιγραφών (Description Logic), επιτρέποντας αυστηρή λογική επεξεργασία και αποδείξεις. Επίσης υποστηρίζει συμπερασματολογία (reasoning) επιτρέποντας τη χρήση Fact, Hermit, Racer κ.α.

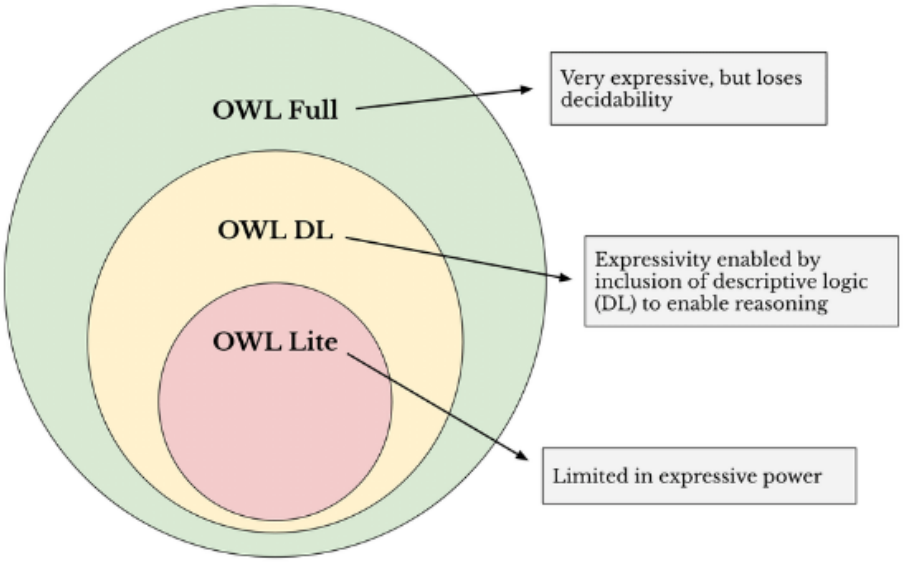
3.4.1 OWL 1

Στο πλαίσιο του συμβιβασμού μεταξύ εκφραστικής ισχύος και αποτελεσματικής υποστήριξης συμπερασματολογίας το W3C ορίζει τρείς διαφορετικές υπογλώσσες της OWL 1.

OWL Full χρησιμοποιεί όλες τις βασικές αρχές της γλώσσας OWL και επιτρέπει τον αυθαίρετο συνδυασμό αυτών των αρχών με το RDF και το RDF Schema. Είναι πλήρως συμβατό με το RDF, τόσο συντακτικά όσο και σημασιολογικά, πράγμα που σημαίνει ότι κάθε έγκυρο έγγραφο RDF είναι επίσης έγκυρο έγγραφο OWL Full, και κάθε έγκυρο συμπέρασμα RDF/RDF Schema είναι επίσης έγκυρο συμπέρασμα OWL Full. Ωστόσο, το OWL Full είναι τόσο ισχυρό που είναι μη αποφασίσιμο, πράγμα που σημαίνει ότι δεν υπάρχει πλήρης ή αποδοτική υποστήριξη για reasoning.

OWL DL (Description Logic) είναι μια υπογλώσσα του OWL Full που εισάγει ορισμένους περιορισμούς για να διασφαλίσει ότι η συλλογιστική μπορεί να γίνει αποτελεσματικά. Συγκεκριμένα περιορίζει τη χρήση των δομικών στοιχείων σε OWL και RDF ώστε να μην μπορούν να εφαρμοστούν μεταξύ τους με αυθαίρετους τρόπους. Επιτρέπει την αποτελεσματική υποστήριξη reasoning όμως χάνεται η πλήρης συμβατότητα με το RDF, αφού δεν είναι όλα τα έγγραφα RDF έγκυρα έγγραφα OWL DL. Αντίθετα, κάθε έγκυρο έγγραφο OWL DL είναι επίσης έγκυρο έγγραφο RDF.

OWL Lite είναι μια ακόμη αυστηρότερη περιορισμένη εκδοχή του OWL DL, που περιορίζει τη γλώσσα σε ένα υποσύνολο των κατασκευαστικών στοιχείων. Για παράδειγμα, το OWL Lite εξαιρεί τις καταμετρημένες κλάσεις (enumerated classes), τις δηλώσεις αμοιβαίας εξαίρεσης (disjoint) και την αυθαίρετη αριθμητικότητα (cardinality). Το πλεονέκτημα αυτού του περιορισμού είναι ότι η γλώσσα είναι πιο εύκολη στην κατανόηση για τους χρήστες και πιο εύκολη στην εφαρμογή για τους δημιουργούς εργαλείων. Ωστόσο, το μειονέκτημα είναι ότι η εκφραστική δύναμη της γλώσσας είναι περιορισμένη. Η γλώσσα OWL κάλυψε την ανεπάρκεια σε



Σχήμα 3.9: Τα εκφραστικά επίπεδα των οντολογιών OWL 1.

εκφραστικότητα των RDF και RDFS που περιορίζεται σε μία ιεραρχία κλάσεων και ιδιοτήτων με:

- Τη δυνατότητα για τοπικούς περιορισμούς σε domain και range. Στο RDFS, οι περιορισμοί για domain και range είναι παγκοσμιοί, δηλαδή εφαρμόζονται σε όλα τα αντικείμενα και τις σχέσεις. Με την OWL 1, μπορούμε να ορίσουμε ότι μια ιδιότητα έχει διαφορετικό range ή domain ανάλογα με την κλάση στην οποία εφαρμόζεται.
- Ορίζοντας διακριτές κλάσεις (disjoint classes). Αυτό σημαίνει ότι δύο κλάσεις δεν μπορούν να περιέχουν ταυτόχρονα το ίδιο άτομο.
- Τη δυνατότητα της ένωσης (union), όπου μπορούμε να δημιουργήσουμε νέες κλάσεις που αποτελούνται από τη συνένωση άλλων κλάσεων.
- Εισαγωγή της έννοιας ισοδυναμίας (equivalence) για κλάσεις και ιδιότητες, κάτι που δεν

υποστηρίζεται στο RDFS. Η ισοδυναμία επιτρέπει τον ορισμό ότι δύο κλάσεις ή δύο ιδιότητες είναι ισοδύναμες και αναφέρονται στην ίδια έννοια.

- Ορισμό περιορισμών καρδιναλιότητας για τις ιδιότητες, κάτι που δεν υποστηρίζεται στο RDFS. Η καρδιναλιότητα ορίζει τον αριθμό των τιμών που μπορεί να έχει μια ιδιότητα για μια οντότητα. Μπορούμε να καθορίσουμε ότι μια ιδιότητα πρέπει να έχει ακριβώς, τουλάχιστον ή το πολύ έναν συγκεκριμένο αριθμό τιμών.
- Υποστήριξη για ειδικά χαρακτηριστικά των ιδιοτήτων, όπως μεταβατικές (transitive), αντιστροφές (inverse), συμμετρικές (symmetric).

3.4.2 OWL 2

Κλάσεις (Classes) Μία κλάση περιγράφει κάποια έννοια. Οι κλάσεις παρέχουν έναν μηχανισμό αφαίρεσης για την ομαδοποίηση πόρων με παρόμοια χαρακτηριστικά. Υπάρχουν αρκετοί τρόποι περιγραφής μίας κλάσης ο πρώτος είναι δηλώνοντας την με το μοναδικό αναγνωριστικό κλάσης (URI). Παράδειγμα ex:Person a owl:Class . Μπορούμε όμως να περιγράψουμε μία κλάση μέσα από περιορισμούς ιδιότητας (property restriction), με πλήρη απαρίθμηση (enumeration) όλων των ατόμων που αποτελούν τα στιγμιότυπά της, αλλά και με την τομή (intersection) /ένωση (union) /συμπλήρωμα (complement) δύο ή περισσότερων περιγραφών κλάσης. Για παράδειγμα εάν ορίσουμε το εύρος μιας ιδιότητας ex:gender να είναι Male ή Female τότε ο συμπεραστής (reasoner) θα δημιουργήσει μια ανώνυμη κλάση που αντιπροσωπεύει την ένωση αυτών των δύο κλάσεων.
Ιδιότητες/Ρόλοι (Properties/Roles) Μία ιδιότητα περιγράφει μία σχέση μεταξύ δύο στοιχείων. Η OWL διαθέτει δύο βασικές κατηγορίες ιδιοτήτων. Object properties συνδέουν άτομα με άτομα και Datatype properties που συνδέουν άτομα με τιμές δεδομένων.
Άτομα/Στιγμιότυπα (Individuals/Instances of classes)Κάθε κλάση OWL συνδέεται με ένα σύνολο ατόμων, που ονομάζεται επέκταση κλάσης (class extension). Τα άτομα στην επέκταση κλάσης ονομάζονται στιγμιότυπα της κλάσης.

Βασικές έννοιες: **Αξιώματα** (axioms): οι βασικές δηλώσεις που εκφράζει μια οντολογία OWL. **Οντότητες** (entities): στοιχεία που χρησιμοποιούνται για να αναφερθούμε σε αντικείμενα του πραγματικού κόσμου. **Εκφράσεις** (expressions): συνδυασμοί των οντοτήτων ώστε να δημιουργήσουμε πολύπλοκες περιγραφές από τις βασικές.

3.4.3 Χαρακτηριστικά

Κάποια από τα χαρακτηριστικλά που προστέθηκαν στην OWL 2 είναι:

- **Κλειδιά (Keys)** Επιτρέπουν τον μοναδικό προσδιορισμό ατόμων βάσει συγκεκριμένων ιδιοτήτων. Για παράδειγμα, μπορούμε να ορίσουμε ότι το ”ΑΦΜ” είναι μοναδικό για κάθε άτομο, διασφαλίζοντας ότι δεν υπάρχουν δύο διαφορετικά άτομα με το ίδιο ΑΦΜ.

- **Αλυσίδες Ιδιοτήτων (Property Chains)** Επιτρέπουν τον ορισμό σύνθετων σχέσεων μεταξύ ιδιοτήτων. Για παράδειγμα, αν κάποιος είναι παιδί ενός ατόμου και αυτό το άτομο είναι παιδί κάποιου άλλου, τότε μπορεί να συναχθεί ότι το πρώτο άτομο είναι εγγόνι του τρίτου.

- **Πλουσιότεροι τύποι δεδομένων και περιοχές δεδομένων (Richer Datatypes, Data Ranges)** Η OWL 2 υποστηρίζει πιο σύνθετες περιοχές τιμών για ιδιότητες δεδομένων, όπως αριθμητικά διαστήματα ή περιορισμούς σε συγκεκριμένες τιμές, επιτρέποντας μεγαλύτερη ακρίβεια στην αναπαράσταση δεδομένων.

- **Περιορισμοί ποσοτικότητας με ειδικές συνθήκες (Qualified Cardinality Restrictions)** Επιτρέπουν τον έλεγχο της ποσότητας ατόμων που μπορούν να συνδέονται μέσω μιας ιδιότητας, αλλά και τον περιορισμό αυτής της ποσότητας σε συγκεκριμένες κατηγορίες. Για παράδειγμα, μπορούμε να ορίσουμε ότι ένας φοιτητής μπορεί να έχει το πολύ δύο καθηγητές επιβλέποντες.

- **Ασύμμετρες, Ανακλαστικές και Διαχωρισμένες Ιδιότητες (Asymmetric, Reflexive, and Disjoint Properties)** Επιτρέπει τον ορισμό ασύμμετρων (π.χ. ”είναι ανώτερος από”), ανακλαστικών (π.χ. ”σχετίζεται με τον εαυτό του”) και διαχωρισμένων ιδιοτήτων (δύο ιδιότητες που δεν μπορούν να ισχύουν ταυτόχρονα για τα ίδια άτομα).

- **Βελτιωμένες δυνατότητες σχολιασμού (Enhanced Annotation Capabilities)** Η OWL 2 παρέχει πιο ευέλικτους τρόπους για την προσθήκη σχολίων, μεταδεδομένων και τεκμηρίωσης σε οντολογίες, επιτρέποντας καλύτερη κατανόηση και διαχείριση των εννοιών τους.

3.4.4 Βασικές Υποθέσεις

Υπόθεση Ανοιχτού Κόσμου (Open World Assumption)
Στις περισσότερες γλώσσες που χρησιμοποιούν την υπόθεση του κλειστού κόσμου (CWA), υποθέτουμε ότι όλα όσα είναι γνωστά για το σύστημα βρίσκονται ήδη στη βάση δεδομένων. Ωστόσο, η OWL δημιουργήθηκε ως γλώσσα για να προσθέσει σημασιολογία στο Διαδίκτυο, οπότε οι σχεδιαστές της επέλεξαν την υπόθεση του ανοιχτού κόσμου (OWA). Η υπόθεση του ανοιχτού κόσμου σημαίνει ότι δεν μπορούμε να υποθέσουμε πως κάτι δεν υπάρχει απλώς και μόνο επειδή δεν βρίσκεται στην οντολογία μας. Το Διαδίκτυο είναι ένα ανοιχτό σύστημα και η πληροφορία μπορεί να υπάρχει σε κάποια πηγή δεδομένων που δεν έχει ακόμη ενσωματωθεί

στην οντολογία μας. Επομένως, δεν μπορούμε να συμπεράνουμε ότι κάποια πληροφορία δεν υπάρχει, εκτός αν δηλωθεί ρητά ότι δεν υπάρχει. Με άλλα λόγια, το γεγονός ότι κάτι δεν έχει δηλωθεί ως αληθές δεν σημαίνει ότι είναι ψευδές απλώς υποθέτουμε ότι η γνώση αυτή δεν έχει προστεθεί ακόμη στη γνωσιακή βάση.

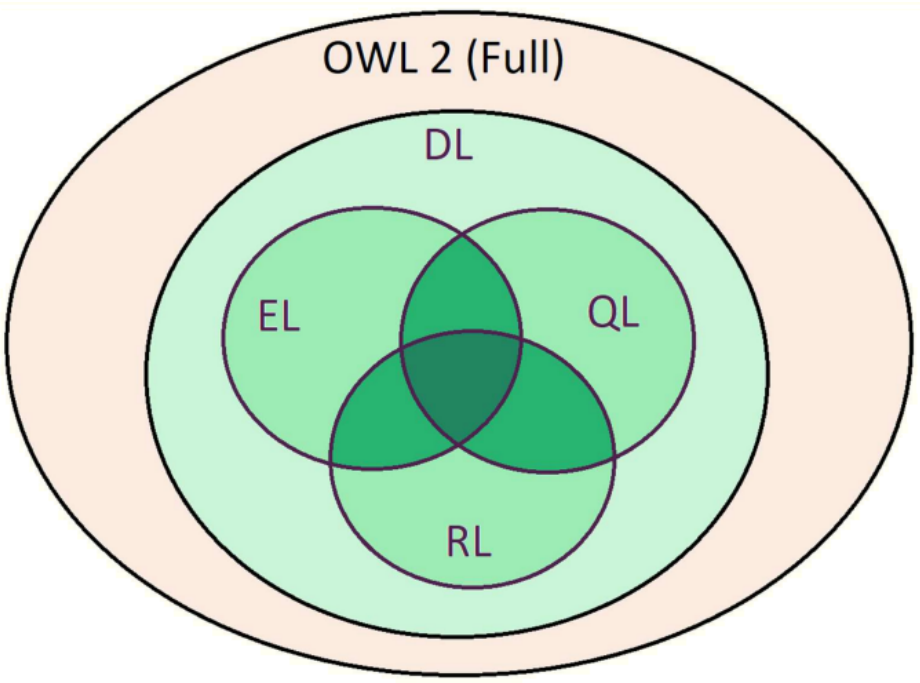
Υπόθεση Μη-μοναδικού Ονόματος (Non-unique Naming Assumption)
Δύο διαφορετικά ονόματα ή αναγνωριστικά δεν θεωρούνται απαραίτητα ότι αναφέρονται σε διαφορετικές οντότητες. Στην OWL, αν δεν δηλωθεί ρητά ότι δύο ονόματα είναι διακριτά (με χρήση του owl:differentFrom), μπορεί να αναφέρονται στο ίδιο αντικείμενο. Για παράδειγμα, τα "Queen Elizabeth", "The Queen" και "Elizabeth Windsor" μπορεί όλα να αναφέρονται στο ίδιο άτομο. Αυτό επιτρέπει μεγαλύτερη ευελιξία στην ενσωμάτωση δεδομένων από διαφορετικές πηγές, αλλά απαιτεί προσεκτικό σχεδιασμό για την αποφυγή ασαφειών.

3.4.5 OWL 2 Profiles

Υπο-γλώσσες (συντακτικά υποσύνολα) τα οποία προσφέρουν διαφορετικό βαθμό εκφραστικότητας και πολυπλοκότητας. Κάθε προφίλ προσφέρει σημαντικά πλεονεκτήματα σε συγκεκριμένα σενάρια εφαρμογής. Τρία διαφορετικά προφίλ: OWL 2 EL, OWL 2 QL, και OWL 2 RL. Κάθε προφίλ έχει οριστεί ως συντακτικός περιορισμός των προδιαγραφών της OWL 2, δηλ. ως υποσύνολο των δομικών στοιχείων που μπορούν να χρησιμοποιηθούν σε μία οντολογία, και το καθένα είναι πιο περιοριστικό από την OWL DL. Κάθε ένα από τα προφίλ προσφέρει διαφορετικές υπολογιστικές δυνατότητες ή/και δυνατότητες εφαρμογής.[13]

EL: Από το "Existential Logic" ή "Expressive Logic". Αναφέρεται στη δυνατότητα υποστήριξης υπαρξιακών περιορισμών, δηλαδή σχέσεων που καθορίζουν ότι κάποια ιδιότητα πρέπει να υπάρχει χωρίς να χρειάζεται πλήρης περιγραφή του αντικειμένου. Σχεδιασμένο για οντολογίες που χρειάζονται αποτελεσματική ταξινόμηση και διαχείριση μεγάλου αριθμού κλάσεων και σχέσεων, όπως ιατρικά ή βιολογικά δεδομένα. Επιτρέπει τη χρήση σύνθετων ιεραρχιών και σχέσεων, αλλά περιορίζει πιο εκφραστικές δυνατότητες (π.χ. disjoint unions). Κατάλληλη για εφαρμογές όπου απαιτούνται πολύ μεγάλες οντολογίες.

QL: Από το "Query Language". Βελτιστοποιημένο για εφαρμογές που χρησιμοποιούν μεγάλες βάσεις δεδομένων και επιθυμούν αποδοτικά ερωτήματα σε λογικό χρόνο. Είναι ιδιαίτερα κατάλληλο για ερωτήματα SQL, υποστηρίζοντας αποδοτικά ερωτήματα χωρίς πλήρη ταξινόμηση οντολογιών. Επιτρέπει συνδυαστικά ερωτήματα που πρέπει να απαντηθούν σε συγκεκριμένο χρόνο χρησιμοποιώντας τεχνολογία σχεσιακών βάσεων δεδομένων. Κατάλληλη για εφαρμογές με σχετικά μικρές οντολογίες με μεγάλο αριθμό ατόμων και ανάγκη άμεσης πρόσβασης στα δεδομένα μέσω σχεσιακών ερωτημάτων (π.χ., SQL).



Σχήμα 3.10: OWL 2 Profiles.

RL: Από το ”Rule Language”. Σχετίζεται με τη δυνατότητα χρήσης κανόνων (rules) για εξαγωγή λογικών συμπερασμάτων και είναι βελτιστοποιημένο για επεξεργασία μέσω συστημάτων βασισμένων σε κανόνες. Εστιάζει στην ενίσχυση της λογικής επεξεργασίας μέσω κανόνων (rulebased reasoning). Κατάλληλο για εφαρμογές όπου χρειάζεται αποδοτική επεξεργασία με κανόνες, μπορεί να χρησιμοποιηθεί με κανόνες RDFS και σε συστήματα επαγωγικών μηχανών (inference engines) χωρίς περίπλοκη λογική. Κατάλληλες για εφαρμογές με σχετικά μικρές οντολογίες για την οργάνωση μεγάλου αριθμού ατόμων και όπου χρειάζεται η απευθείας λειτουργία σε δεδομένα με τη μορφή RDF τριάδων.

3.5 Σύγκριση OWL – OOP

Στους παρακάτω πίνακες γίνεται μία σύγκριση OWL/RDF με τις γλώσσες OOP που είμαστε περισσότερα εξοικειωμένοι με σκοπό την καλύτερη κατανόηση των δυνατοτήτων και του τρόπου λειτουργίας της OWL

Object-Oriented Languages	OWL
Κλάσεις και Στιγμιότυπα	
Οι κλάσεις θεωρούνται τύποι για τα στιγμιότυπα.	Οι κλάσεις θεωρούνται ως σύνολα ατόμων (individuals).
Κάθε στιγμιότυπο έχει μία και μόνο κλάση ως τύπο του. Οι κλάσεις δεν μπορούν να μοιράζονται στιγμιότυπα.	Κάθε άτομο μπορεί να ανήκει σε πολλές κλάσεις.
Τα στιγμιότυπα δεν μπορούν να αλλάξουν τύπο κατά την εκτέλεση.	Η ένταξη σε μία κλάση μπορεί να αλλάξει κατά την εκτέλεση.
Η λίστα των κλάσεων είναι πλήρως γνωστή κατά τη μεταγλώττιση και δεν μπορεί να αλλάξει.	Οι κλάσεις μπορούν να δημιουργηθούν και να αλλάξουν δυναμικά.
Οι μεταγλωττιστές χρησιμοποιούνται κατά το στάδιο της κατασκευής και εντοπίζουν σφάλματα.	Χρησιμοποιούνται reasoners για ταξινόμηση και έλεγχο συνέπειας κατά την εκτέλεση ή κατά τη δημιουργία του μοντέλου.
Ιδιότητες, Χαρακτηριστικά και Τιμές	
Οι ιδιότητες ορίζονται τοπικά σε μία κλάση (και στις υποκλάσεις της μέσω κληρονομικότητας).	Οι ιδιότητες είναι ανεξάρτητες οντότητες που μπορούν να υπάρχουν χωρίς συγκεκριμένες κλάσεις.
Τα στιγμιότυπα μπορούν να έχουν τιμές μόνο για τις συνδεδεμένες ιδιότητες. Οι περιορισμοί τύπου χρησιμοποιούνται για έλεγχο τύπου.	Τα στιγμιότυπα μπορούν να έχουν οποιαδήποτε τιμή για οποιαδήποτε ιδιότητα. Οι περιορισμοί τύπου και περιοχές ορισμού χρησιμοποιούνται για έλεγχο τύπου.
Οι κλάσεις κωδικοποιούν μεγάλο μέρος της σημασιολογίας και της συμπεριφοράς μέσω διαδικασιών και μεθόδων.	Οι κλάσεις κάνουν τη σημασία τους ρητή μέσω δηλώσεων OWL. Δεν επισυνάπτεται διαδικαστικός κώδικας.
Οι κλάσεις μπορούν να αποκρύψουν τα μέλη τους με ιδιωτική πρόσβαση.	Όλα τα τμήματα ενός αρχείου OWL/RDF είναι δημόσια και μπορούν να συνδεθούν από οπουδήποτε.
(CWA)Αν δεν υπάρχει αρκετή πληροφορία για να αποδειχθεί μια δήλωση ως αληθής, τότε θεωρείται ψευδής.	(OWA)Αν δεν υπάρχει αρκετή πληροφορία για να αποδειχθεί μια δήλωση ως αληθής, τότε μπορεί να είναι είτε αληθής είτε ψευδής.

Πίνακας 3.1: Σύγκριση OWL – OOP.

Object-Oriented Languages	OWL
Ρόλος στη Διαδικασία Σχεδίασης	
Κάποιες γενικές API μοιράζονται μεταξύ εφαρμογών. Λίγα (ή καθόλου) UML διαγράμματα είναι κοινόχρηστα.	Το RDF και το OWL έχουν σχεδιαστεί εξαρχής για τον Ιστό. Τα μοντέλα τομέα μπορούν να κοινοποιούνται διαδικτυακά.
Τα μοντέλα τομέα σχεδιάζονται ως μέρος της αρχιτεκτονικής λογισμικού.	Τα μοντέλα τομέα σχεδιάζονται για την αναπαράσταση γνώσης και την ολοκλήρωση πληροφοριών.
UML, Java, C κ.λπ. είναι ώριμες τεχνολογίες με εκτενή υποστήριξη.	Ο Σημασιολογικός Ιστός είναι μία αναδυόμενη τεχνολογία με ορισμένα open-source εργαλεία και περιορισμένο αριθμό εμπορικών προμηθευτών.

Πίνακας 3.2: Σύγκριση OWL – OOP.

3.6 SHACL

SHACL (Shapes Constraint Language)

Η SHACL (Shapes Constraint Language) είναι μια γλώσσα που έχει σχεδιαστεί από το W3C για την επαλήθευση και τον έλεγχο των δεδομένων που είναι δομημένα σύμφωνα με το μοντέλο RDF. Αποτελεί εργαλείο-κλειδί για τη διασφάλιση της ποιότητας, της συνέπειας και της ορθότητας των δεδομένων. Η λειτουργία του βασίζεται στην περιγραφή των επιθυμητών χαρακτηριστικών και δομών των δεδομένων μέσω RDF γραφημάτων. Οι κανόνες και οι περιορισμοί που ορίζονται μπορούν να περιλαμβάνουν τύπους δεδομένων, υποχρεωτικά πεδία, τιμές πεδίων, καθώς και σχέσεις μεταξύ διαφορετικών οντοτήτων. Η SHACL υποστηρίζει διάφορους τύπους περιορισμών, όπως ιδιοτήτων (property constraints) και κόμβων (node constraints). Η γλώσσα προσφέρει μεγάλη ευελιξία και επεκτασιμότητα, επιτρέποντας τη δημιουργία σύνθετων σχημάτων που αντικατοπτρίζουν την πολυπλοκότητα των δεδομένων πραγματικού κόσμου. Επιπλέον υποστηρίζει τη χρήση προσαρμοσμένων συναρτήσεων μέσω της SPARQL, ενισχύοντας τη δυνατότητα έκφρασης πιο εξειδικευμένων κανόνων. Ένα από τα βασικά πλεονεκτήματα της SHACL είναι η ενσωμάτωσής της σε υπάρχουσες τεχνολογίες RDF και SPARQL. Αυτό την καθιστά συμβατή με πλήθος εργαλείων και πλατφορμών που χρησιμοποιούνται για την επεξεργασία και την αποθήκευση δεδομένων RDF. Είναι σχετικά νεότερη από τις άλλες τεχνολογίες που περιγράφονται εδώ. Ωστόσο, καλύπτει ένα ουσιαστικό κενό στη στοίβα αρχιτεκτονικής του Σημασιολογικού Ιστού και αποκτά ολοένα και μεγαλύτερη απήχηση στον κόσμο της ανάπτυξης μεγάλων εταιρικών συστημάτων. [14]
Αν και περιορισμοί για τα δεδομένα μπορούν επίσης να καθοριστούν χρησιμοποιώντας Περι-

γραφική Λογική ή SWRL, που είναι πιο υψηλού επιπέδου και λίγο πιο εύχρηστα υπάρχουν δύο λόγοι για τους οποίους η SHACL είναι απαραίτητη:

1. Η ανάγκη καθορισμού περιορισμών που δεν υπόκεινται στην Υπόθεση Ανοιχτού Κόσμου (OWA) και στη Μονοτονική Συλλογιστική. Η υπόθεση ότι τα δεδομένα «ίσως υπάρχουν κάπου εκεί έξω» δεν αρκεί. Για να διασφαλίσουμε την ακεραιότητα των δεδομένων, πρέπει να μπορούμε να ενεργοποιούμε προειδοποιήσεις όταν τα απαιτούμενα δεδομένα δεν υπάρχουν, οπότε πρέπει να χρησιμοποιήσουμε την Υπόθεση Κλειστού Κόσμου (CWA).

2. Το γεγονός ότι στα πραγματικά δεδομένα εμφανίζονται ασυνέπειες και σφάλματα με αποτέλεσμα ο μηχανισμός συλλογιστικής να χαρακτήρισε την οντολογία σας ως μη έγκυρη. Σε μικρά παραδείγματα αυτό δεν αποτελεί πρόβλημα ωστόσο, όταν ασχολούμαστε με Big Data, όπου υπάρχουν δεκάδες χιλιάδες, εκατομμύρια ή και περισσότερα άτομα, η συχνότητα εμφάνισης κακών δεδομένων μπορεί να είναι τεράστια.

Για παράδειγμα έστω οτι θέλουμε κάθε αντίγραφο του ”Employee” να έχει έναν και μόνο έναν αριθμό κοινωνικής ασφάλισης (ssn). Αν αυτό οριζόταν ως αξίωμα DL, τότε το αξίωμα δεν θα ενεργοποιούνταν ποτέ για υπαλλήλους που δεν είχαν ssn λόγω της Αρχής του Ανοιχτού Κόσμου (OWA). Από την άλλη πλευρά, αν ένας υπάλληλος είχε κατά λάθος δύο τιμές ssn, τότε ολόκληρη η οντολογία θα ήταν ασυνεπής μέχρι να αφαιρεθεί μία τιμή. Με την SHACL μπορούμε να χειριστούμε και τις δύο περιπτώσεις χωρίς να τίθεται όλη η οντολογία ασυνεπή, απλώς καταγράφοντας προειδοποιήσεις. [15]

«ex:EmployeeShape a sh:NodeShape;» Ορίζει το σχήμα (shape) ως NodeShape, το οποίο σημαίνει ότι θα εφαρμόζεται σε μεμονωμένους κόμβους RDF, και το ονομάζει ex:EmployeeShape.
«sh:targetClass ex:Employee;» Προσδιορίζει ότι το σχήμα ισχύει για πόρους που ανήκουν στην κλάση ex:Employee.
«sh:property [...]» Ορίζει περιορισμούς για μια συγκεκριμένη ιδιότητα του ex:Employee.
«sh:path ex:ssn;» Καθορίζει την ιδιότητα που θα εφαρμοστούν.
«sh:minCount 1;» και «sh:maxCount 1;» Περιορίζει την διότητα ssn να εμφανίζεται τουλάχιστον μία αλλά και μόνο μία φορά.
«sh:pattern “”;» Καθορίζει τη μορφή που θα πρέπει να έχει σε regex.
«sh:severity sh:Warning ;» Αυτή η γραμμή καθορίζει τη βαρύτητα του περιορισμού στην προκειμένη περίπτωση αν το SSN δεν ταιριάζει με το μοτίβο, θα εκδοθεί μια προειδοποίηση.

3.7 SWRL

Όπως όλα τα συστήματα κανόνων, το SWRL αποτελείται από μια αριστερή πλευρά (που ονομάζεται προϋπόθεση - antecedent) και μια δεξιά πλευρά (που ονομάζεται αποτέλεσμα - consequent).

```
ex:EmployeeShape
    a sh:NodeShape ;
    sh:targetClass ex:Employee ;
    sh:property [
        sh:path ex:ssn ;
        sh:minCount 1 ;
        sh:maxCount 1 ;
        sh:datatype xsd:string ;
        sh:pattern "^[0-9]{3}-[0-9]{2}-[0-9]{4}$" ;
        sh:message "Το SSN πρέπει να έχει τη μορφή NNN-NN-NNNN, όπου το N είναι ένας αριθμός."
        sh:severity sh:Warning ;
    ] .
```

Σχήμα 3.11: Παράδειγμα Shacl σε turtle σύνταξη.

Οι δύο πλευρές χωρίζονται από ένα βέλος: ->. Κάθε έκφραση σε έναν κανόνα SWRL διαχωρίζεται με το σύμβολο ∧. Το αποτέλεσμα του κανόνα ενεργοποιείται εάν και μόνο εάν ικανοποιούνται όλες οι εκφράσεις της προϋπόθεσης. Επειδή η προϋπόθεση μπορεί να ικανοποιηθεί πολλές φορές, αυτό σημαίνει ότι οι κανόνες SWRL μπορούν να εκτελούν επαναλήψεις. Θα ενεργοποιηθούν για κάθε συνδυασμό τιμών που μπορεί να ικανοποιήσει την προϋπόθεση. Όλες οι παράμετροι (μεταβλητές που λειτουργούν ως μπαλαντέρ και δεσμεύονται δυναμικά καθώς εκτελείται ο κανόνας) προηγούνται από το σύμβολο ?.

Οι εκφράσεις SWRL αποτελούνται από 3 τύπους:

1. Εκφράσεις κλάσεων: τα ονόματα των κλάσεων ακολουθούμενα από παρενθέσεις με μια παράμετρο. Customer(?c) θα δεσμεύσει το ?c σε μια εμφάνιση της κλάσης Customer και (υποθέτοντας ότι το υπόλοιπο της προϋπόθεσης ικανοποιείται) θα επαναληφθεί για κάθε εμφάνιση της κλάσης Customer.

2. Εκφράσεις ιδιοτήτων: τα ονόματα ιδιοτήτων, ακολουθούμενα από παρενθέσεις και δύο παραμέτρους, η πρώτη για το άτομο που ελέγχεται και η δεύτερη για να δεσμεύσει την τιμή αυτής της ιδιότητας για το συγκεκριμένο άτομο. Επειδή τα άτομα μπορεί να έχουν περισσότερες από μία τιμές για μια ιδιότητα, αυτό μπορεί επίσης να δημιουργήσει επαναλήψεις, όπου οι κανόνες θα εκτελούνται για κάθε τιμή ιδιότητας για κάθε άτομο.

3. Ενσωματωμένες συναρτήσεις (built-in functions): το SWRL περιλαμβάνει διάφορες ενσωματωμένες συναρτήσεις για μαθηματικούς ελέγχους, ελέγχους συμβολοσειρών κ.λπ. οι οποίες μπορούν να βρεθούν εδώ:https://www.w3.org/Submission/SWRL/. Όλες οι ενσωματωμένες συναρτήσεις SWRL προηγούνται από το πρόθεμα swrlb. Π.χ., η μαθηματική

ενσωματωμένη συνάρτηση swrlb:greaterThan(?np, 1) θα επιτύχει εάν η τιμή του ?np είναι μεγαλύτερη από 1.

3.8 SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) Η ανακάλυψη πληροφοριών στον Σημασιολογικό Ιστό περιλαμβάνει διαφορετικές μεθόδους για την εύρεση δεδομένων RDF, ανάλογα με το πόσα γνωρίζουμε για τη δομή και τη θέση των δεδομένων. Οι τρεις κύριες προσεγγίσεις είναι η πλοήγηση, η αναζήτηση και τα ερώτηματα. Τα ερωτήματα είναι η πιο ακριβής μέθοδος, που επιτρέπει δομημένες και σύνθετες ερωτήσεις που χρησιμοποιούν επίσημη σύνταξη αλλά απαιτεί γνώση της διατύπωσης ερωτημάτων και της δομής δεδομένων RDF. Η εκτέλεση ερωτημάτων στον Σημασιολογικό Ιστό απαιτεί μια γλώσσα που αναγνωρίζει το RDF ως τη θεμελιώδη σύνταξη.
Υπάρχουν πολλές γλώσσες ερωτημάτων RDF, όπως η RDQL (RDF Data Query Language) και η SeRQL (Sesame RDF Query Language), παρ'ολα αυτά η SPARQL υπερέχει λόγω της τυποποίησής της από το W3C, της ευρείας υποστήριξής από την κοινότητα και του μεγάλου αριθμού διαθέσιμων endpoints. Ένα τελικό σημείο (endpoint), είναι μια υπηρεσία (όχι απαραίτητα διαδικτυακή) που δέχεται και επεξεργάζεται ερωτήματα SPARQL, επιστρέφοντας αποτελέσματα σε διαφορετικές μορφές ανάλογα με τον τύπο του ερωτήματος. Τα endpoints που είναι προσβάσιμα μέσω HTTP πρέπει να ακολουθούν το πρωτόκολλο SPARQL (https://www.w3.org/TR/2013/REC-sparql11-protocol-20130321/ /).Η επίσημη προδιαγραφή της γλώσσας SPARQL μπορεί να βρεθεί εδώ (https://www.w3.org/TR/sparql11-overview/).
Επίσης η SPARQL είναι τόσο γλώσσα ερωτημάτων όσο και πρωτόκολλο. Το πρωτόκολλο καθορίζει τον τρόπο επικοινωνίας ενός SPARQL client, όπως ενός προγράμματος περιήγησης, με ένα SPARQL endpoint, όπως το http://dbpedia.org/sparql, τόσο σε αφηρημένο επίπεδο όσο και μέσω συγκεκριμένης υλοποίησης βασισμένης στο WSDL 2.0. Σε αυτήν την ενότητα θα εστιάσουμε στην SPARQL ως γλώσσα ερωτημάτων. [16]

3.8.1 Δομη ερωτημάτων

Δηλώσεις προθεμάτων που χρησιμεύουν στη συντομογραφία των χρησιμοποιούμενων URIs.
PREFIX foo: <http:.//example.com/resources>

…

Δηλώσεις συνόλων δεδομένων που δηλώνουν τους γράφους RDF που χρησιμοποιούνται.
FROM …

Πρόταση αποτελέσματος που προσδιορίζει το αποτέλεσμα που θέλουμε να επιστρέψει το ερώτημα.

SELECT …
Μοτίβο ερωτήματος που ορίζει το ζητούμενο από το υποκείμενο σύνολο δεδομένων.
WHERE
…
Μετατροπείς ερωτημάτων που μετασχηματίζουν τα αποτελέσματα αναζήτησης.
ORDER BY …
Ένα τελικό σημείο SPARQL (endpoint) δέχεται ερωτήματα SPARQL και επιστρέφει τα αντίστοιχα αποτελέσματα μέσω του πρωτοκόλλου HTTP.

3.8.2 Δυνατότητες

Η SPARQL υποστηρίζει τέσσερις διαφορετικές μορφές ερωτημάτων:

- SELECT – Παρόμοιο με το SELECT στη SQL, το SELECT ανακτά RDF όρους (κενά nodes, IRIs ή literals) και τους αντιστοιχίζει σε μεταβλητές με βάση το μοτίβο γραφήματος (π.χ., την ενότητα WHERE). Τα αποτελέσματα επιστρέφονται σε μορφή πινάκων αλλά δεν αποτελούν μέρος ενός RDF γραφήματος.

- CONSTRUCT – Επιτρέπει τη μετατροπή των δεσμευμένων μεταβλητών σε ένα νέο RDF γράφημα, αρκεί κάθε τριπλέτα να είναι έγκυρη (π.χ., χωρίς literals σε θέση υποκειμένου ή κατηγόρου). Αυτή η μορφή ερωτήματος είναι χρήσιμη για τη μετατροπή δεδομένων μεταξύ RDF γραφημάτων ή οντολογιών OWL. Τα αποτελέσματα μπορούν να αποθηκευτούν ή να συνδυαστούν με άλλα RDF γραφήματα.

- ASK – Χρησιμοποιείται για να ελέγξει αν υπάρχει ένα συγκεκριμένο γράφημα, επιστρέφοντας μόνο μια τιμή boolean (true/false). Είναι χρήσιμο για γρήγορες ερωτήσεις, αποφεύγοντας πολύπλοκα και απαιτητικά ερωτήματα SELECT ή CONSTRUCT.

- DESCRIBE – Επιστρέφει ένα RDF γράφημα που καθορίζεται από τον επεξεργαστή (endpoint), με ελάχιστη καθοδήγηση από τον χρήστη. Είναι χρήσιμο όταν ο χρήστης δεν γνωρίζει τη δομή των δεδομένων, επιτρέποντας τη συλλογή βασικών πληροφοριών. Ωστόσο, χρησιμοποιείται λιγότερο συχνά από τις άλλες μορφές.

Η πρώτη επίσημη έκδοση του προτύπου δημοσιεύτηκε τον Ιανουάριο του 2008, ενώ η πιο πρόσφατη έκδοση είναι η SPARQL 1.1, η οποία δημοσιεύτηκε τον Μάρτιο του 2013. Μερικές απο τις λειτουργίες αναφέρονται παρακάτω:

Μορφότυπα: Η SPARQL υποστηρίζει τέσσερα διαδεδομένα μορφότυπα ανταλλαγής δεδομένων: Extensible Markup Language(XML), JavaScript Object Notation(JSON), comma-separated values(CSV) και tab-separated values(TSV). Με αυτόν τον τρόπο τα αποτελέσματά μας από τα

ερωτήματα που είναι σύνολα αντιστοίχησης μεταβλητών σε δεδομένα RDF και αναπαρίστανται ως πίνακες μπορούμε να τα εκφράσουμε σε μηχαναγνώσιμη μορφή.

Συναθροίσεις(Aggregates): Είναι συναρτήσεις που χρησιμοποιούνται για την εκτέλεση υπολογισμών σε σύνολα τιμών, βασικοί συναθροιστές είναι οι εξής:
COUNT: Μετράει τον αριθμό των τιμών.
SUM: Υπολογίζει το άθροισμα.
AVG: Υπολογίζει τον μέσο όρο.
MIN: Βρίσκει την ελάχιστη τιμή.
MAX: Βρίσκει τη μέγιστη τιμή.
GROUP CONCAT: Συνενώνει τιμές σε μια συμβολοσειρά.
SAMPLE: Επιστρέφει μια τυχαία τιμή.

Υποερωτήματα: Τα υποερωτήματα αποτελούν έναν τρόπο να συμπεριλάβουμε ερωτήματα μέσα σε άλλα ερωτήματα. Κάτι τέτοιο είναι χρήσιμο σε διάφορες περιπτώσεις, όπως στον περιορισμό του αριθμού των αποτελεσμάτων από κάποια υποέκφραση μέσα στο ερώτημα. Στη SPARQL τα υποερωτήματα εκτελούνται πρώτα και τα αντίστοιχα αποτελέσματα προσφέρονται στο εξωτερικό ερώτημα.

Update: Λειτουργίες ενημέρωσης χρησιμοποιώντας τις παρακάτω λέξεις κλειδιά για αλλαγές σε RDF γράφους που βρίσκονται σε ένα αποθετήριο.
INSERT: Προσθήκη νέων δεδομένων.
DELETE: Διαγραφή υπαρχόντων δεδομένων.
LOAD: Φόρτωση δεδομένων από ένα URI.
CLEAR: Διαγραφή όλων των δεδομένων από ένα γράφημα ή ολόκληρο το σύνολο δεδομένων.
CREATE: Δημιουργία ενός νέου, κενού γραφήματος.
DROP: Διαγραφή ενός γραφήματος ή ολόκληρου του συνόλου δεδομένων.
COPY/MOVE: Αντιγραφή/Μετακίνηση δεδομένων από ένα γράφημα σε ένα άλλο.
ADD: Προσθήκη δεδομένων από ένα γράφημα σε ένα άλλο.

Προσθήκη πολλαπλών γράφων: Η RDF επιτρέπει την οργάνωση των δεδομένων σε default graphs (προεπιλεγμένα γραφήματα) και named graphs (ονομασμένα γραφήματα). Το default graph περιέχει δεδομένα που δεν ανήκουν σε κάποιο συγκεκριμένο ονομασμένο γράφημα, ενώ τα named graphs είναι υποσύνολα του συνόλου δεδομένων που ταυτοποιούνται με ένα URI. Με την χρήση των FROM και FROM NAMED η SPARQL επιτρέπει την προσθήκη συγκεκριμένων γραφημάτων στο ερώτημα, όπου η FROM προσθέτει default graphs και η FROM NAMED προσθέτει named graphs. Η αναζήτηση σε πολλαπλά γραφήματα γίνεται με τη χρήση της λέξης-

κλειδί GRAPH, η οποία επιτρέπει την ανάκτηση δεδομένων από συγκεκριμένα ονομασμένα γραφήματα.

```
PREFIX ex: <http://www.example.org/>
SELECT …
FROM ex:g1
FROM ex:g3
FROM NAMED ex:g1
FROM NAMED ex:g2
WHERE
… A …
GRAPH ex:g2
… B …
GRAPH ?graph
… C …
```

Αν ένα URI αναφέρεται τόσο σε δήλωση NAMED όσο σε δήλωση FROM NAMED, αυτό σημαίνει ότι οι αντίστοιχες τριάδες στο αποθετήριο συμμετέχουν τόσο στον εξ ορισμού όσο και σε έναν επώνυμο γράφο. Τα μοτίβα που βρίσκονται στη θέση ... A ... θα προσπαθήσουν να ταιριάξουν με τριάδες του εξ ορισμού γράφου που αποτελείται από τα URI ex:g1 και ex:g3, ενώ τα μοτίβα που βρίσκονται στη θέση ... C ... θα προσπαθήσουν να ταιριάξουν με τριάδες όλων των επώνυμων γράφων του αποθετηρίου που αντιστοιχούν στα URI ex:g1, ex:g2.

FILTER: Η λέξη FILTER επιτρέπει τη χρήση δυαδικών συνθηκών για το φιλτράρισμα μη επιθυμητών αποτελεσμάτων. Η SPARQL υποστηρίζει πλήθος συναρτήσεων όπως λογικές, μαθηματικές, συγκριτικές, ελέγχου, εκτίμησης, συνθήκης.

UNION/OPTIONAL: Με την χρήση UNION μπορούμε να ενώσουμε δύο μοτίβα και να ζητήσουμε τριάδες οι οποίες να ικανοποιούν και τα δύο.Ενώ με την χρήση του OPTIONAL αν το μοτίβο τριάδων που ακολουθεί αποτύχει να ταυτιστεί με κάποια τριάδα στον γράφο, οι μεταβλητές που συμμετέχουν σε αυτό παραμένουν χωρίς τιμή (unbound).

Μετατροπείς(modifiers): Χρησιμοποιούνται για να ελέγξουν και να τροποποιήσουν τα αποτελέσματα ενός ερωτήματος. Οι πιο συνηθισμένοι μετατροπείς είναι τα LIMIT, OFFSET, ORDER BY, και DISTINCT.

Inferencing:Με την SPARQL μπορούμε να εκτελέσουμε ερωτήματα αναζήτησης και ανάκτησης δεδομένων πάνω σε συστήματα που υποστηρίζουν συμπερασμοτολογία (inferencing) για να εξάγουμε λογικά συμπεράσματα από τα δεδομένα.

Federated query: Υποστηρίζει αναζήτηση σε απομακρυσμένα datasets χρησιμοποιώντας τη λέξη κλειδί SERVICE για να στέλνει ερωτήματα σε remote SPARQL endpoints. Αυτές οι δυνατότητες καθιστούν τη SPARQL ισχυρή και ευέλικτη για την ανάκτηση και διαχείριση δεδομένων από πολλαπλές πηγές, είτε είναι τοπικές είτε απομακρυσμένες. Στο παρακάτω ερώτημα αρχικά βρίσκουμε στο DBpedia τον τόπο γέννησής του Προέδρου Ομπάμα και το αντίστοιχο Geonames ID. Στη συνέχεια, πηγαίνουμε στο Geonames για να βρούμε όλα τα άλλα Geonames IDs που αντιστοιχούν σε κατοικημένες περιοχές εντός 10 μιλίων από τον τόπο γέννησης του Προέδρου Ομπάμα. Με αυτήν τη λίστα των Geonames IDs, στη συνέχεια πηγαίνουμε στην απογραφή του 2000 και βρίσκουμε το μέσο εισόδημα για κάθε περιοχή.

```
PREFIX geo: <http://franz.com/ns/allegrograph/3.0/geospatial/>
PREFIX geonames: <http://sws.geonames.org/>
PREFIX dbpedia_rsrc: <http://dbpedia.org/resource/>
PREFIX dbpedia_onto: <http://dbpedia.org/ontology/>
PREFIX dbpedia_prop: <http://dbpedia.org/property/>
PREFIX census: <tag:govshare.info,2005:rdf/census/>
PREFIX census_samp: <tag:govshare.info,2005:rdf/census/details/samp/>
SELECT distinct ?censusplace ?income {
  dbpedia_rsrc:Barack_Obama dbpedia_onto:birthPlace ?birthplace .
  ?birthplace dbpedia_prop:hasGeonamesID ?geonamesresource .
  SERVICE <https://localhost:10000/catalogs/demos/repositories/geonames>
            { ?geonamesresource geonames:isAt5 ?location .
              ?otherplace geo:inCircleMiles (geonames:isAt5 ?location 10) .
              ?otherplace geonames:feature_code "PPL" .
              ?geonamesresource geonames:feature_code "PPL" .
          SERVICE <https://localhost:10000/catalogs/demos/repositories/census>
            { ?censusplace dbpedia_prop:hasGeonamesID ?otherplace .
              ?censusplace census:details ?detail .
              ?detail census_samp:population15YearsAndOverWithIncomeIn1999 ?d .
              ?d census_samp:medianIncomeIn1999 ?income .}}}
```

Σχήμα 3.12: Sparql ερώτημα χρησιμοποιώντας Service.

Κεφάλαιο 4

Μοντελοποίηση Προγράμματος Σπουδών

Εισαγωγή

Σε αυτό το κεφάλαιο θα περιγράψουμε τον τρόπο μοντελοποίησης του Προγράμματος Σπουδών (ΠΣ). Αρχικά, θα πρέπει να ορίσουμε τις βασικές οντότητες που θα περιέχει το μοντέλο, καθώς και τις συσχετίσεις μεταξύ τους. Θα χρειαστούμε κλάσεις όπως Μάθημα (Course), Φοιτητής (Student) και Πρόγραμμα Σπουδών (Program Study). Τα βασικά δεδομένα των μαθημάτων προς μοντελοποίηση βρίσκονται στη σελίδα: https://www.iee.ihu.gr/udg_courses/, όπου για κάθε μάθημα παρουσιάζονται οι σχετικές πληροφορίες ξεχωριστά. Στην παρούσα εργασία θα εξαιρέσουμε τα δύο τελευταία πεδία, που αφορούν τους συντονιστές και τους διδάσκοντες.

Σχήμα 4.1: Πληροφορίες μαθημάτων προς μοντελοποίηση.

Σύμφωνα με αυτά τα δεδομένα, είναι σημαντικό να αναζητήσουμε στο διαδίκτυο υπάρχουσες οντολογίες και λεξιλόγια που δομούν αυτές τις σχέσεις και καλύπτουν τις ανάγκες μας. Ορι-

σμένες χρήσιμες πηγές για την αναζήτηση οντολογιών και λεξιλογίων βρίσκονται στη σελίδα: https://www.w3.org/standards/semanticweb/, όπου παρουσιάζονται με τις επίσημες προδιαγραφές και οδηγίες καθώς πολλά από αυτά αποτελούν μέρος των προτύπων του W3C (World Wide Web Consortium):
Linked Open Vocabularies (LOV) (https://lov.linkeddata.es) Ένα καλά συντηρημένο αποθετήριο λεξιλογίων και οντολογιών που χρησιμοποιούνται στον Σημασιολογικό Ιστό.
W3C OWL Ontologies (https://www.w3.org/2001/sw/wiki/Ontologies) Μια συλλογή από οντολογίες που προτείνονται ή χρησιμοποιούνται σε έργα του W3C.
The Open Ontology Repository (OOR) (http://ontolog.cim3.net/cgi-bin/wiki.pl?OpenOntologyRepository) Μια συνεργατική πλατφόρμα για την κοινή χρήση και διαχείριση οντολογιών.

Αν και υπάρχουν αρκετές οντολογίες για την περιγραφή σχέσεων μαθημάτων σε ένα πρόγραμμα σπουδών, αυτή που μας καλύπτει περισσότερο είναι η Schema.org. Με τον όρο schema.org/Course περιγράφονται τα μαθήματα, ενώ το schema.org/EducationalOccupationalProgram περιγράφει ένα πρόγραμμα σπουδών που προσφέρεται από ιδρύματα για την επίτευξη ενός συγκεκριμένου αποτελέσματος, συνήθως ενός πτυχίου. Το Schema.org χρησιμοποιεί κυρίως το RDFS και στοιχεία του OWL Lite, αλλά δεν αποτελεί πλήρη OWL οντολογία, σχεδιάστηκε για την απλή και ευχρηστία σήμανση περιεχομένου στο διαδίκτυο. Το μοντέλο που θα δημιουργήσουμε θα βασίζεται στην OWL, ώστε να αξιοποιήσουμε τις δυνατότητες του αυτόματου συμπερασμού για την εξαγωγή νέας γνώσης. Η νέα γνώση θα αφορά τον εντοπισμό των οφειλόμενων μαθημάτων για κάθε φοιτητή. Στο τρέχον πρόγραμμα σπουδών, ένας φοιτητής, για να ολοκληρώσει τις σπουδές του, πρέπει να έχει περάσει 33 κοινά υποχρεωτικά μαθήματα (32 + πτυχιακή εργασία), 5 υποχρεωτικά μαθήματα ομάδας και 8 μαθήματα επιλογής (η πρακτική άσκηση ισοδυναμεί με 2 μαθήματα επιλογής). Θα γίνει μια προσπάθεια μοντελοποίησης αυτών των προϋποθέσεων στο ΠΣ, χρησιμοποιώντας τις δυνατότητες της OWL για συμπερασματολογία και τον ορισμό κανόνων στο μοντέλο. Αν και θα μπορούσαμε να δημιουργίσουμε από μόνοι μας την οντολογία με κάποια εύκολη προς τον χρήστη σύνταξη όπως η turtle υπαρχουν αρκτετα εργαλεια για τον σκοπο αυτο οπως το Protégé, TopBraid Composer και το OntoStudio.

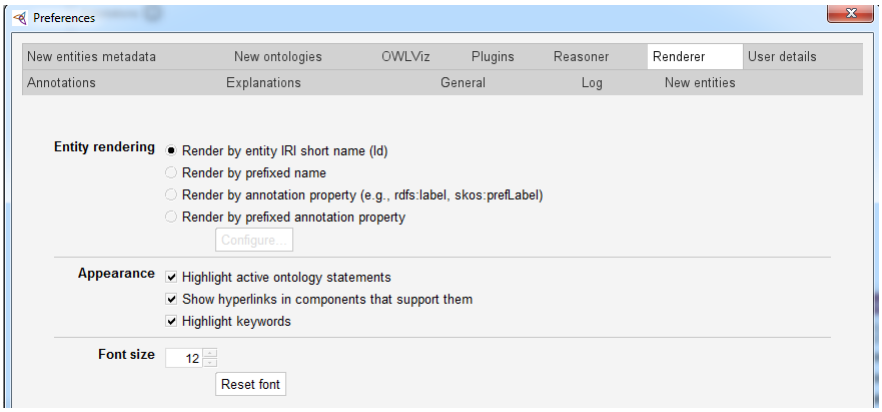
4.1 Protege

Το πιο δημοφιλές και ευρέως χρησιμοποιούμενο εργαλείο ειναι το Protégé που προσφέρει μια γραφική διεπαφή (GUI) διευκολύνοντας τη δημιουργία και την επεξεργασία οντολογιών.Είναι ένα εργαλείο ανοιχτού κώδικα που αναπτύχθηκε από το Πανεπιστήμιο του Στάνφορντ και μπορεί να βρεθεί στην σελίδα https://protege.stanford.edu/ .Οι οντολογίες που δημιουργούνται με το Protégé μπορούν να χρησιμοποιηθούν για την αναπαράσταση γνώσης σε διάφορους τομείς, όπως η ιατρική, η βιοπληροφορική, οι επιστήμες των υπολογιστών, κ.ά.Υποστηρίζει πλήρως

τη γλώσσα OWL και ειναι συμβατό με RDF,RDFS επιτρέποντας την αναπαράσταση σύνθετων σχέσεων μεταξύ οντοτήτων και την κατασκευή εκφραστικών οντολογιών.Διαθέτει μία βιβλιοθήκη επεκτάσεων όπου μπορεί να βρεί κανείς ανοιχτού κώδικα και εμπορικά plug-ins για πιο εξειδικευμένες ανάγκες.Η πλατφόρμα περιλαμβάνει εργαλεία για τη δοκιμή και την αξιολόγηση των οντολογιών, όπως η ανίχνευση ασυμβατοτήτων και η εκτέλεση λογικών ερωτημάτων. Δυνατότητα εξαγωγής οντολογιών σε διάφορες μορφές, όπως RDF/XML, Turtle και OWL, καθιστώντας τις συμβατές με άλλα εργαλεία και συστήματα.Υποστηρίζει τη διασύνδεση δεδομένων με τη φιλοσοφία του Linked Data επιτρέποντας την εύκολη σύνδεση και την ανάλυση δεδομένων από διαφορετικές πηγές. Καθώς είναι ικανό να διαχειριστεί και οντολογίες μεγάλου μεγέθους, κάνοντάς το κατάλληλο για επαγγελματικές και βιομηχανικές εφαρμογές. [17]

4.1.1 Δημιουργία οντολογίας

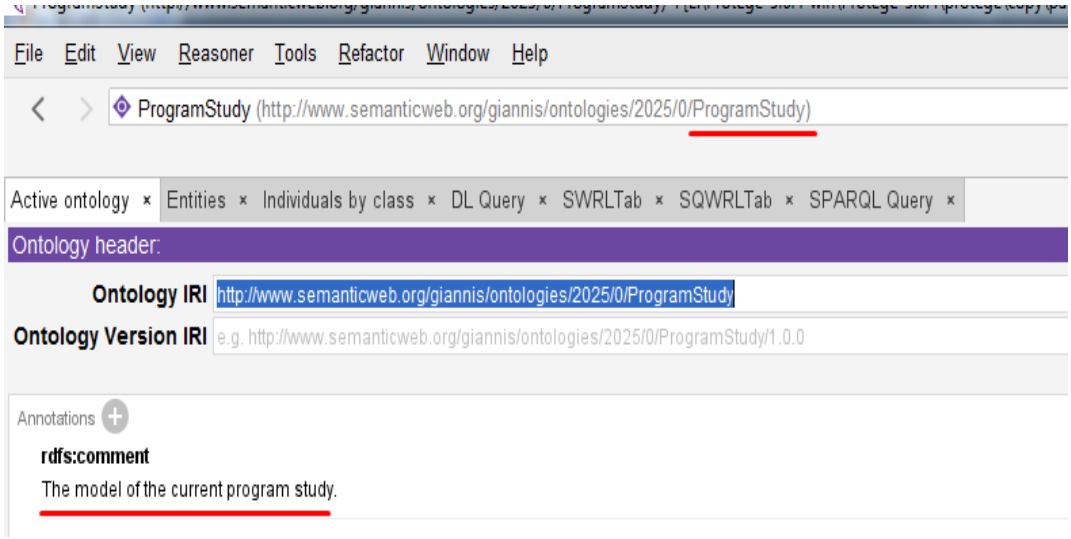
Το protége ανοίγει πάντα με μια νέα οντολογία χωρίς τίτλο και το IRI έχει την μορφή http://www.semanticweb.org/yourname/ontologies/2025/4/untitled-ontology-12. Άπο την καρτέλα Active Ontology που παρέχει πληροφορίες επισκόπησης μπορούμε να αλλάξουμε το τελευταίο μέρος σε ProgramStudy σχήμα 4.2. [15] [18]
Το Protégé αναγνοριζει κάθε κλάση, στιγμιότυπο, ιδιότητα αντικειμένου, ιδιότητα δεδομένων, ιδιότητα σχολιασμού ή κανόνα ως οντότητα οπότε είναι σημαντικό να ορίσουμε κάποιες ρυθμίσεις για το πως θα μας εμφανίζονται τα ονόματα των οντοτήτων. Ο όρος όνομα στην OWL μπορεί να αναφέρεται σε δύο διαφορετικές έννοιες είτε στο τελευταίο μερος του Iri είτε στην ιδιότητα σχολιασμού (συνήθως rdfs:label) που χρησιμοποιείται για να είναι πιο φιλικό προς το χρήστη. Απο την καρτέλα File>Preferences>Renderer, οι πρώτες δύο επιλογές στην ενότητα entity rendering αναφέρονται στο Iri με την επιλογή εμφάνισης του προθέματος ενώ οι άλλες δύο στην ιδιότητα σχολιασμού, για το συγκεκριμένο μοντέλο επιλέγουμε την πρώτη σχήμα 4.1.



Σχήμα 4.2: Renderer tab.

Προσθέτουμε σχόλιο σημείωσης για την οντολογία από το παράθυρο Ontology header> Annotations

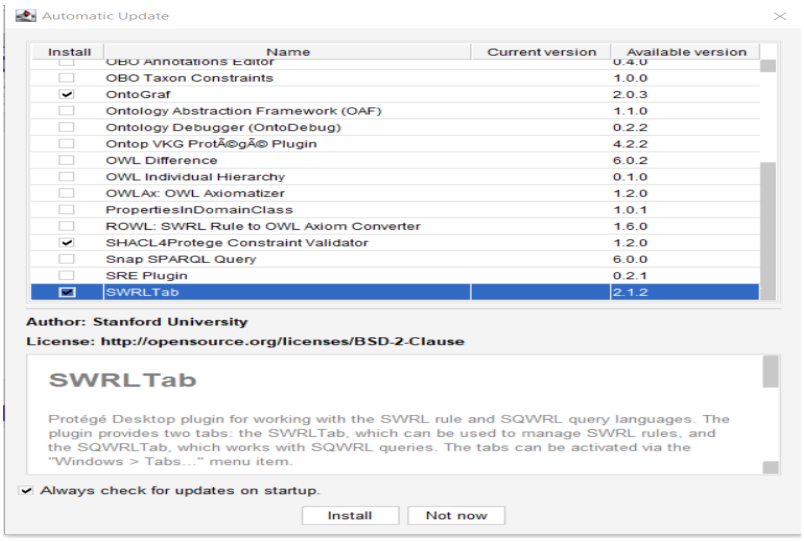
και αυτόματα το protégé θα προσθέσει τη γραμμή κώδικα 'rdfs:comment' μαζί με το σχόλιο μας σχήμα 4.2.



Σχήμα 4.3: Επισκόπηση.

4.1.2 Plug-ins

Ανοίγοντας το Protégé εμφανίζεται ένα παράθυρο με προτεινόμενα plugins για να εγκαταστήσουμε αλλιώς μπορούμε να το βρούμε και από File>Check for plugins.
Αυτά που χρειαζόμαστε είναι:
Pellet reasoner, αν και υπάρχουν κάποιοι προεγκατεστημένοι συμπεραστές και διάφοροι άλλοι διαθέσιμοι ως πρόσθετα δεδομένου ότι πρόκειται να γράψουμε κανόνες SWRL ο Pellet έχει την καλύτερη υποστήριξη για το SWRL.
OntoGraf, παρέχει υποστήριξη για διαδραστική απεικόνιση στις σχέσεις των οντολογιών. Υποστηρίζει διάφορες διατάξεις για την αυτόματη οργάνωση της δομής μιας οντολογίας . Υποστηρίζονται διαφορετικές σχέσεις όπως υποκλάση, άτομο, ιδιότητες αντικειμένου, τομέας/εύρος και ισοδυναμία. Καθώς επιτρέπει και την οπτικοποίηση με φίλτρα στις σχέσεις και τους κόμβους αυτούς.
Shacl4Protege, παρέχει μια καρτέλα, την οποία μπορούμε να ανοίξουμε και να επεξεργαστούμε αρχεία SHACL, και να επικυρώσουμε την οντολογία που έχει φορτωθεί στο Protégé.
SwrlTab, παρέχει ένα περιβάλλον ανάπτυξης για ανάπτυξη κανόνων SWRL και ερωτήματα SQWRL σχήμα 4.3.

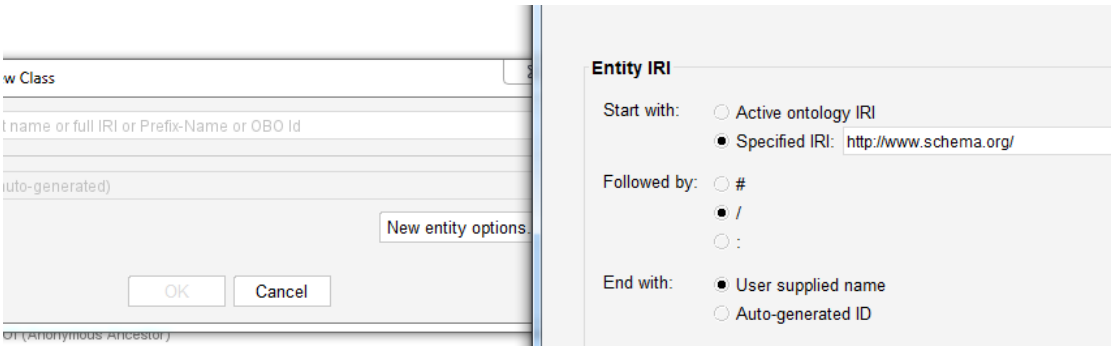


Σχήμα 4.4: Protégé plugins.

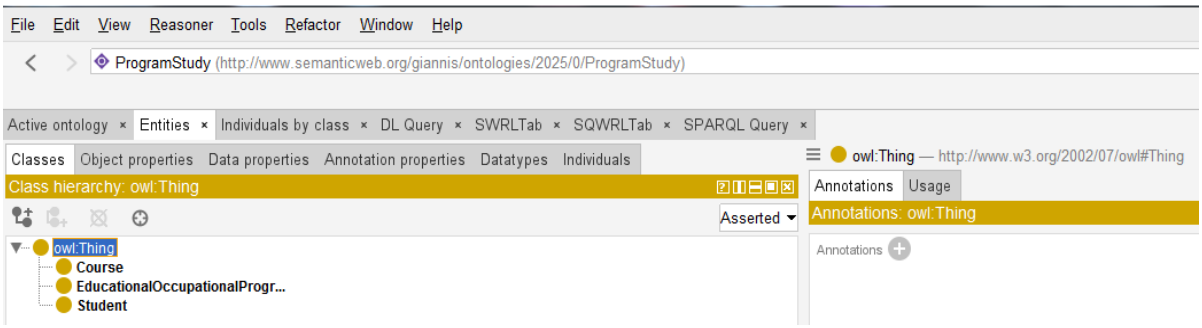
4.1.3 Classes

Τα κύρια δομικά στοιχεία μιας οντολογίας OWL είναι οι κλάσεις. Στο Protege 5, η επεξεργασία των κλάσεων μπορεί να γίνει στην καρτέλα Entities και στις υποκαρτέλες της. Όταν την επιλέγουμε, η προεπιλογή θα πρέπει να είναι το Class hierarchy όπως φαίνεται στο Σχήμα 4.4. Όλες οι κενές οντολογίες περιέχουν μια κλάση που ονομάζεται owl:Thing και αντιπροσωπεύει το σύνολο που περιέχει όλα τα άτομα (individuals). Οι κλάσεις OWL είναι σύνολα ατόμων και επομένως είναι υποκλάσεις του owl:Thing. Απο την καρτέλα Entities επιλέγουμε Classes και σιγουρευόμαστε ότι το owl:Thing είναι επιλεγμένο. Πατάμε το εικονίδιο Add Subclass για την δημιουργία υποκλάσεων στην επιλεγμένη κλάση και θα εμφανιστεί ένα παράθυρο διαλόγου με τίτλο Create a new class με ένα πεδίο για το όνομα της νέας κλάσης όπου βάζουμε Student. Απο κατω συμπληρώνεται αυτόματα το IRI που θα χαρακτηρίζει μοναδικά τη συγκεκριμένη κλάση.

Δεν υπάρχουν υποχρεωτικές συμβάσεις ονομασίας για τις οντότητες παρ' όλα αυτά θα χρησιμοποιήσουμε την πρακτική camelback για τις κλάσεις και τα στιγμιότυπα. Το schema.org παρέχει μία κλάση EducationalOccupationalProgram η οποία περιγράφει προγράμματα που προσφέρονται από ιδρύματα για την απόκτηση πτυχίου και θα την χρησιμιποιήσουμε για την δημιουργία της αντίστοιχης κλάσης. Πρώτα φτιάχνουμε ένα prefix από την καρτέλα Ontology prefixes με prefix name: "sch:" και prefix: http://www.schema.org/. Επιλέγουμε πάλι Add Subclass στο owl:Thing απλά αυτήν τη φορά από το παράθυρο Create a new Class επιλέγουμε New entity options για να καθορίσουμε το IRI όπως φαίνεται στο σχήμα 4.5 και στη συνέχεια να ορίσουμε το όνομα της κλάσης EducationalOccupationalProgram.Με τον ίδιο τρόπο δημιουργούμε την κλάση Course και η διεπαφή χρήστη θα πρέπει να μοιάζει με το σχήμα 4.5. [19]

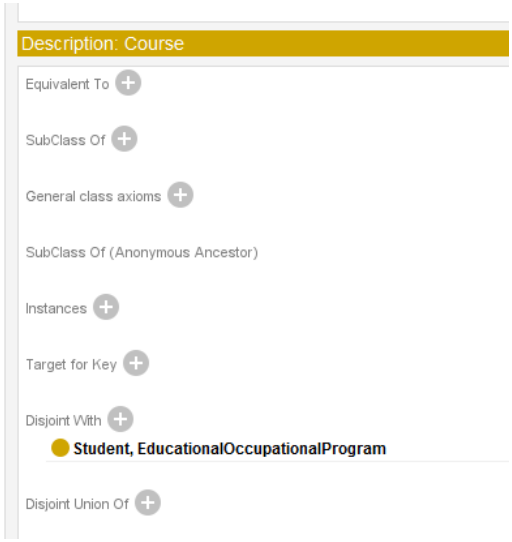


Σχήμα 4.5: Εισαγωγή IRI από τον χρήστη για δημιουργλία κλάσης.



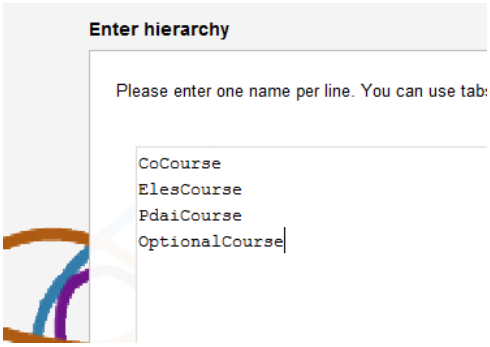
Σχήμα 4.6: Διεπαφή χρήστη.

Έχοντας προσθέσει τις κλάσεις θα προσθέσουμε disjoint ιδιότητα για να γίνουν ασύνδετες μεταξύ τους. Δηλαδή, καμία οντότητα δεν μπορεί να ανήκει σε περισσότερες από μία από αυτές τις κλάσεις. Στην ορολογία της θεωρίας συνόλων, η τομή αυτών των τριών κλάσεων είναι το κενό σύνολο: owl:Nothing. Στην OWL οι κλάσεις απο προεπιλογή μπορούν να αλληλεπικαλύπτονται δηλαδή δεν είναι διακριτές και επιτρέπει έτσι την πολλαπλή κληρονομικότητα για τη μοντελοποίηση δεδομένων σε αντίθεση με τα περισσότερα αντικειμενοστραφή μοντέλα. Επιλέγουμε μία από τις κλάσεις και στην καρτέλα Description πατάμε στο Disjoint with. Από το παράθυρο που θα εμφανιστεί στην καρτέλα Class hierarchy βρίσκουμε και επιλέγουμε τις άλλες δύο κλάσεις σχήμα 4.6. Αυτόματα δημιουργούνται και τα disjoint στις άλλες κλάσεις.
Οι υποκλάσεις που θα έχει το μοντέλο μας είναι: CoCourse για τα μαθήματα που είναι υποχρεωτικά για όλους τους φοιτητές προκειμένου να ολοκληρώσουν τις σπουδές τους.
ElesCourse για τα μαθήματα που ανήκουν στην ομάδα Ηλεκτρονικών και Ενσωματωμένων Συστημάτων (ΗΛΕΣ).
PdaiCourse για τα μαθήματα που ανήκουν στην ομάδα Προγραμματισμού Δεδομένων και Τεχνητής Νοημοσύνης (ΠΔΤΝ).
OptionalCourse για τα μαθήματα που είναι επιλογής για όλους τους φοιτητές.
Οπότε για να τις δημιουργίσουμε ελέγχουμε από το File>Preferences>New entities να είναι επιλεγμένα τα “Active ontology IRI” , “#” , “User supplpied name”. Επιλέγοντας την κλάση Course



Σχήμα 4.7: Disjoint classes.

επιλέγουμε Tools>Create class hierarchy και μπορούμε να φτιάξουμε πολλές υποκλάσεις ταυτόχρονα σχήμα 4.7.



Σχήμα 4.8: Course subclasses.

4.1.4 Object properties

Οι ιδιότητες αντιπροσωπεύουν σχέσεις. Υπάρχουν τρεις τύποι ιδιοτήτων ιδιότητες αντικειμένου που συσχετίζουν αντικείμενα μεταξύ τους οι ιδιότητες τύπου δεδομένων για σχέσεις μεταξύ ενός αντικειμένου και ενός τύπου δεδομένων όπως xsd:string ή xsd:dateTime και οι ιδιότητες σχολιασμού. Οι ιδιότητες σχολιασμού έχουν συνήθως τύπους δεδομένων ως τιμές, αν και μπορούν να έχουν αντικείμενα. Μία ιδιότητα σχολιασμού είναι συνήθως μεταδεδομένα, όπως ένα σχόλιο ή μια ετικέτα. Στη σελίδα https://www.iee.ihu.gr/udg_courses/ μπορούμε να βρούμε όλη την πληροφορία που θέλουμε να μοντελοποιήσουμε και αφορά τα μαθήματα. Στο σχήμα 4.9 φαίνεται ένα παράδειγμα για τις ιδιότητες που πρέπει να δημιουργηθούν για κάθε μά-

θημα ξεχωριστά εκτώς από τα πεδία που αφορούν καθηγητές: ‘Συντονιστής’,’Διδάσκοντες’. Από αυτά η μόνη ιδιότητα που χρειάζεται να είναι αντικειμένου είναι για τα προαπαιτούμενα μαθήματα. Βέβαια θα προσθέσουμε και μία ακόμη που θα συνδέει μαθήματα θεωρίας με τα μαθήματα εργαστηρίων.

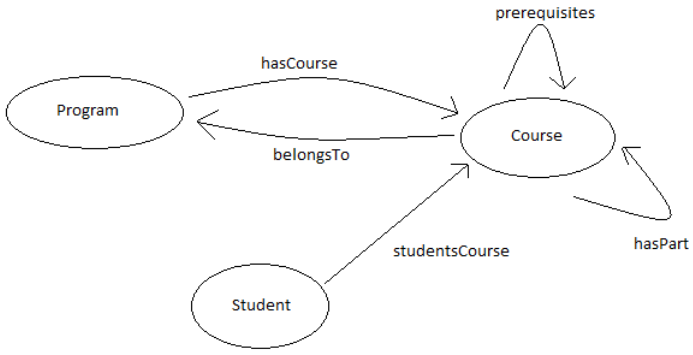
Ενσωματωμένα Συστήματα

Γενικά

- Κωδικός Μαθήματος: **1602**
- Εξάμηνο: **6ο**
- Τύπος Μαθήματος: **Επιστημονικής Περιοχής - Ανάπτυξης Δεξιοτήτων (ΕΠ-ΑΔ)**
- Είδος Μαθήματος: **Υποχρεωτικό (ΥΠ)**
- Γνωστική Περιοχή: **Ενσωματωμένα - Υπολογιστικά Συστήματα (ΕΥΣ)**
- Διδασκαλία Θεωρίας: **4 ώρες/εβδομάδα**
- Διδασκαλία Εργαστηρίου: **2 ώρες/εβδομάδα**
- Πιστωτικές μονάδες ECTS: **6**
- Γλώσσα διδασκαλίας και Εξετάσεων: **Ελληνικά, Αγγλικά**
- Προτεινόμενα προαπαιτούμενα μαθήματα: **(1204) Σχεδίαση Ψηφιακών Συστημάτων, (1102) Δομημένος Προγραμματισμός, (1502) Μικροελεγκτές**
- Συντονιστής: **Παπαδοπούλου Μαρία**
- Διδάσκοντες: **Παπαδοπούλου Μαρία, Γιακουμής Άγγελος, Χαραλαμπίδης Χαράλαμπος**

Σχήμα 4.9: Παράδειγμα μαθήματος με τα δεδομένα για μοντελοποίηση.

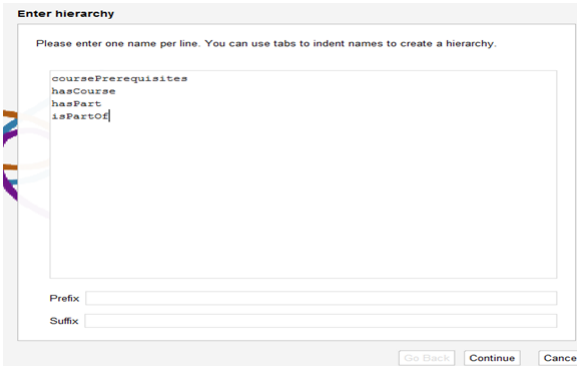
Για τον σκοπό των συγκεκριμένων ιδιοτήτων στο Schema.org/Course μπορούμε να βρούμε ποιες είναι αυτές που περιγράφουν το ζητούμενο που θέλουμε. Καταλήγουμε ότι η coursePrerequisites θα είναι αυτή που θα συνδέει μαθήματα με τα προτεινόμενα προαπαιτούμενα και η hasPart για μαθήματα θεωρίας με μαθήματα εργαστηρίου με την isPartOf ως αντίστροφή της. Επιπλέον θα έχουμε μία ιδιότητα αντικειμένου που συνδέει την κλάση EducationalOccupationalProgram με την κλάση Course. Από το schema.org για την συγκεκρι-μένη κλάση παρέχεται η ιδιότητα hasCourse για τον σκοπό αυτό. Καθώς και μία ιδιότητα αντικειμένου belongsTo αντίστροφη της hasCourse που θα συνδέει Course με EducationalOccupationalProgram. Τέλος μία ιδιότητα αντικειμένου που συνδέει την κλάση Student με Course.



Σχήμα 4.10: Object properties.

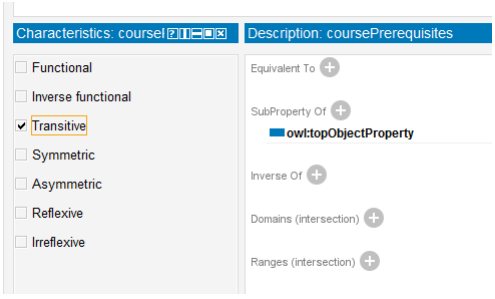
Όπως οι κλάσεις είναι μια υποκατηγορία του owl:Thing οι ιδιότητες είναι μια υπο-ιδιότητα του

owl:topObjectProperty.Από την καρτέλα Entities και Object properties δημιουργούμε τις υποιδιότητες στο owl:topObjectProperty με ονόματα studentsCourse και belongsTo. Για τις υπόλοιπες αλλάζουμε από τις ρυθμίσεις τον τρόπο που θα δοθούν τα IRIs ώστε να μην έχουν από την οντολογία που φτιάχνουμε αλλά από το λεξιλόγιο του Schema.org. File>Preferencies>New entities επιλέγουμε scpecified IRI. Έχοντας επιλεγμένο το owl:topObjectProperty δημιουργούμε ταυτόχρονα τις ιδιότητες από το Tools>Create object property hierarchy.



Σχήμα 4.11: Ιδιότητες αντικειμένων από schema.org.

Από την καρτέλα description της ιδιότητας hasCourse προσθέτουμε την belongsTo ως riverse. Με αυτόν τον τρόπο όταν θα ξεκινήσουμε τον συμπεραστή (reasoner) θα μπορεί από τη σχέση για παράδειγμα: ‘ currentProgram hasCourse course1 ‘ να δημιουργίσει την σχέση ‘ course1 belongsTo currentProgram ‘ προσθέτοντας περισσότερη γνώση στο μοντέλο μας. Εφαρμόζουμε το ίδιο και για τις ιδιότητες hasPart,isPartOf. Στην ιδιότητα coursePrerequisites μπορούμε να τσεκάρουμε την επιλογή transitive σχήμα 4.11 με την οποία ο συμπεραστής έχοντας τις σχέσεις ‘ course3 coursePrerequisites course2 ‘ , ‘ course2 coursePrerequisites course1 ‘ δημιουργεί την σχέση ‘ course3 coursePrerequisites course1 ‘ . Επόμενο βήμα είναι να προσθέσουμε τα Domain και



Σχήμα 4.12: Transitive property.

Ranges για κάθε ιδιότητα δηλαδή να ορίσουμε τα subjects και objects των σχέσεων. Από την

καρτέλα description στα πεδία Domain και Ranges συμπληρώνουμε σύμφωνα με τα παρακάτω.

hasCourse: Domains = EducationalOccupationalProgram, Ranges = Course
belongsTo: Domains = Course, Ranges = EducationalOccupationalProgram
studentsCourse: Domains = Student, Ranges = Course
coursePrerequisites: Domains = Course, Ranges = Course
hasPart: Domain = Course, Ranges = Course
isPartOf: Domain = Course, Ranges = Course

4.1.5 Data properties

Οι ιδιότητες δεδομένων που θα χρειαστούμε για την κλάση Course είναι τα υπόλοιπα πεδία του σχήματος 4.9.Θα χρεισημοποιήσουμε πάλι τις ιδιότητες από τη κλάση schema.org/Course σύμφωνα με τα παρακάτω:

schema.org/courseCode για τον κωδικό κάθε μαθήματος
schema/learningResourceType για τον τύπο μαθήματος με τιμές: General Background (GB), Special Background (SB), General Background - Skills Development (GB-SD), Scientific Area - Skills Development (SA-SD), Scientific Area (SA), Specialization (SP), Specialization - Skills Development (SP-SD), Skills Development (SD).
schema.org/additionalType για το είδος του μαθήματος με τιμές: Compulsory (CO), Optional (OP), Compulsory Optional (CO-OP).
schema.org/about γνωστική περιοχή μαθήματος με τιμές: Generic Knowledge and Skills (GKS), Programming and Algorithms (PA), Embedded Computation Systems (ECS), Electronics (EL), Communications and Networks (CN), Data Management - Artifial Inteligence (DMAI).
schema/timeRequired για τις ώρες διδασκαλίας
schema/repeatFrequency η συχνότητα θα είναι εβδομαδιαία για όλα τα μαθήματα
schema/numberOfCredits για τις πιστωτικές μονάδες ECTS
schema/availableLanguage για τις διαθέσιμες γλώσσες διδασκαλίας και εξετάσεων
schema/courseMode για να δηλώσουμε ότι το μάθημα είναι εγαστήριο

Διαθέσιμε ιδιότητες που θα συσχετιστούν με το πρόγραμμα σπουδών μπορούμε να βρούμε στο schema.org/EducationalOccupationalProgram και είναι:
schema.org/programPrerequisites για τα προαπαιτούμενα εισαγωγής schema.org/hasCredential για τις διδακτικές μονάδες schema.org/timeToComplete για την διάρκεια του προγράμματος

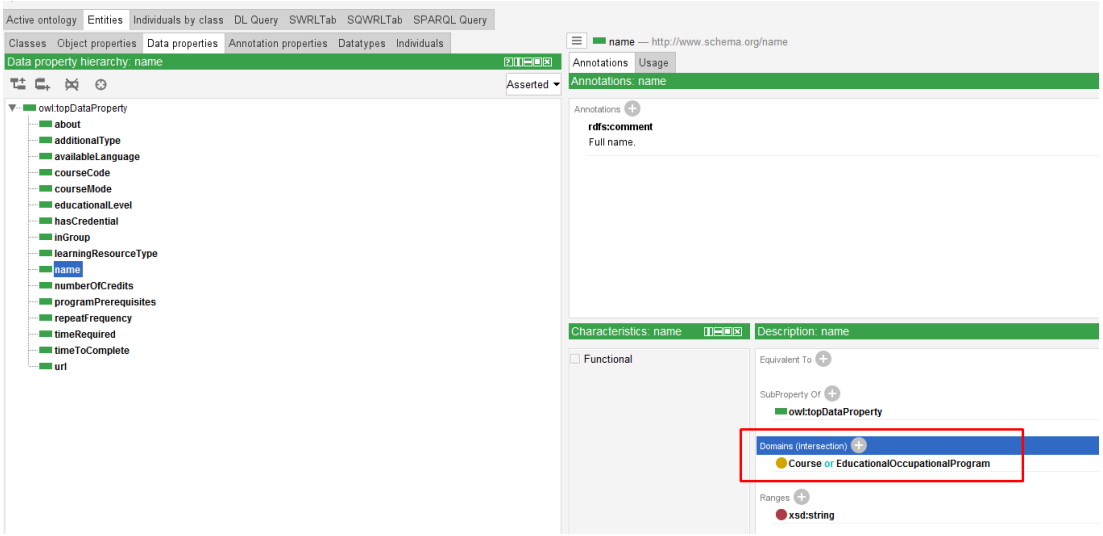
Στην OWL οι ιδιότητες (properties) είναι ανεξάρτητες οντότητες και δεν ανήκουν αποκλειστικά

σε μία κλάση όπως συμβαίνει στον παραδοσιακό αντικειμενοστραφή προ-γραμματισμό. Είναι δηλωμένες ‘ globally ‘ και μπρούν να συσχετιστούν με πολλές κλάσεις. Επομένως οι κοινές ιδιότητες των κλάσεων Course και EducationalOccupationalProgram είναι:

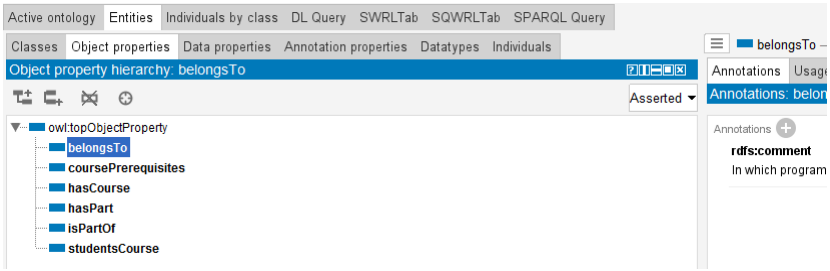
schema.org/name για τα ονόματα μαθημάτων και προγράμματος.
schema.org/educationalLevel για το εξάμηνο κάθε μαθήματος αλλά και για το επίπεδο σπουδών σύμφωνα με το EQF.
schema/url για τις ηλεκτρονικές σελίδες μαθημάτων και προγράμματος σπουδών.

Εκτός από τις διαθέσιμες του λεξιλογίου schema.org θα προσθέσουμε και την:

inGroup για την ομάδα κάθε μαθήματος αλλά και για την ομάδα επιλογής του μαθητή.
Entities>Data properties επιλέγουμε το owl:topDataProperty και τις προσθέτουμε όλες σαν υποιδιότητες.Προσέχουμε τα IRIs να είναι σωστά και από την καρτέλα Annotations συμπληρώνουμε μια σύντομη περιγραφή για κάθε μία. Το Range για όλες τις ιδιότητες είναι xsd:string εκτός της schema/url με Range xsd:anyUri και της numberOfCredits με Range xsd:int. Τα Domain για τις κοινές ιδιότητες ορίζονται μέσα απο το Domain>Class expression editor με τον τελεστή ‘ or ’ ώστε να δηλωθεί η ένωση (union) των κλάσεων διαφορετικά θα δηλωθεί η διατομή των κλάσεων (intersection).



Σχήμα 4.13: Data properties.



Σχήμα 4.14: Object properties.

4.1.6 Προσθήκη κανόνων

Με εγκατεστημένο το πρόσθετο SWRLTab ανοίγουμε την καρτέλα από Window>Tabs> SWRLT όπου θα προσθέσουμε τους κανόνες για το μοντέλο μας. Στόχος είναι πρώτον να συνδέσουμε τον φοιτητή με όλα τα υποχρεωτικά μαθήματα μέσω της ιδιότητας studentCourse ώστε τρέχοντας τον συμπεραστή να μας εμφανίσει όσα δεν έχουν καταχωρηθεί εξαρχής. Δεύτερον για τα μαθήματα επιλογής που έχει περαστεί ένα απο τα δύο μέρη ο συμπεραστής να μας εμφανίσει το μέρος που έχει απομείνει. Αναλυτικά οι κανόνες που θα προσθέσουμε είναι:

1)Σύνδεση των υποχρεωτικών μαθημάτων μέσο της ιδιότητας studentsCourse.

Student(?s) $\wedge sch:Course(?c) \wedge sch:additionalType(?c,?type) \wedge swrlb:equal(?type,$ "Compulsory (CO)") $\rightarrow studentsCourse(?s,?c)$

2)Σύνδεση των υποχρεωτικών εργαστηρίων μέσο της ιδιότητας studentsCourse.

Student(?s) $\wedge sch:Course(?c) \wedge sch:additionalType(?c,?type) \wedge swrlb:equal(?type,$ "Compulsory (CO)") $\rightarrow studentsCourse(?s,?c)$

3)Σύνδεση των υποχρεωτικών μαθημάτων ομάδας μέσο της ιδιότητας studentsCourse.

Student(?s) $\wedge sch:Course(?c) \wedge sch:additionalType(?c,?type) \wedge swrlb:equal(?type,$ "Compulsory Optional (CO-OP)") $\wedge inGroup(?c,?cgroup) \wedge inGroup(?s,?sgroup) \wedge swrlb:equal(?sgroup,?cgroup) \rightarrow studentsCourse(?s,?c)$

4)Σύνδεση των υποχρεωτικών εργαστηρίων ομάδας μέσο της ιδιότητας studentsCourse.

Student(?s) $\wedge sch:Course(?c) \wedge sch:additionalType(?c,?type) \wedge swrlb:equal(?type,$ "Compulsory Optional (CO-OP)") $\wedge inGroup(?c,?cgroup) \wedge inGroup(?s,?sgroup) \wedge swrlb:$

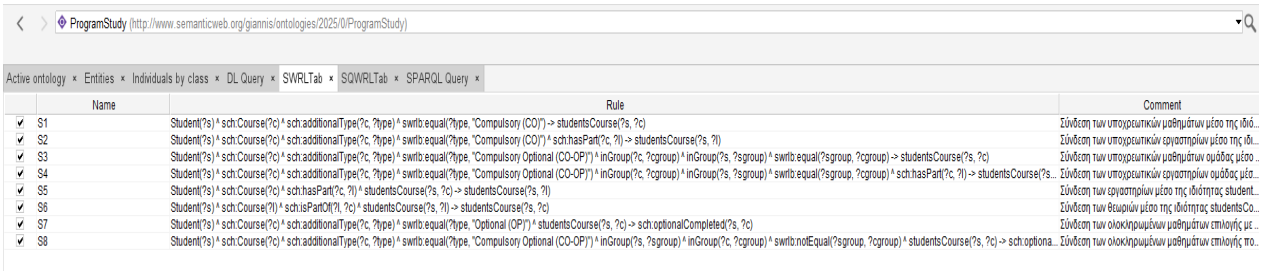
$equal(?sgroup, ?cgroup) \wedge sch : hasPart(?c, ?l) \rightarrow studentsCourse(?s, ?l)$

5)Σύνδεση των εργαστηρίων μέσο της ιδιότητας studentsCourse για όλα τα υπόλοιπα μαθήματα που έχει περαστεί το θεωριτκό μέρος.

Student(?s) $\wedge sch : Course(?c) \wedge sch : hasPart(?c, ?l) \wedge studentsCourse(?s, ?c)$
$\rightarrow studentsCourse(?s, ?l)$

6)Σύνδεση των θεωριών μέσο της ιδιότητας studentsCourse για όλα τα υπόλοιπα μαθήματα που έχει περαστεί το εργαστήριο.

Student(?s) $\wedge sch : Course(?l) \wedge sch : isPartOf(?l, ?c) \wedge studentsCourse(?s, ?l)$
$\rightarrow studentsCourse(?s, ?c)$



Σχήμα 4.15: Swrl κανόνες του μοντέλου.

4.1.7 Restrictions

Στην OWL, μια κλάση μπορεί να οριστεί περιγράφοντας τις διάφορες ιδιότητες και τις τιμές που ισχύουν για όλα τα άτομα της κλάσης. Τέτοιοι ορισμοί ονομάζονται περιορισμοί (restrictions). Υπάρχουν τρεία είδη κλάσεων σύμφωνα με αυτούς τους περιορισμούς:
Primitive κλάσεις που ορίζονται από περιορισμούς που είναι αναγκαίοι (αλλά όχι επαρκείς) για να ισχύουν για οποιαδήποτε άτομα που είναι στιγμιότυπα αυτής της κλάσης ή των υποκλάσεών της.Δηλαδή ακομά και αν ο συμπεραστής βρει κάποιο άτομο που ικανοποιεί αυτές τις συνθήκες δεν επαρκούν για να το ταξινομήση στη κλάση. Αυτό που μπορεί να συμπεραίνει είναι οτι τα άτομα στιγιότυπα μιας κλάσης με αναγκαίους περιορισμούς πρέπει να τους ικανοποιούν.
Defined κλάσεις που ορίζονται από περιορισμούς που είναι τόσο αναγκαίοι όσο και επαρκείς. Όταν ο λογικός μηχανισμός συναντά ένα άτομο που ικανοποιεί όλες τις συνθήκες μιας ορισμένης κλάσης, συμπεραίνει ότι το άτομο είναι στιγμιότυπο αυτής της κλάσης. Μπορεί επίσης

να χρησιμοποιήσει τις συνθήκες που έχουν οριστεί για τις κλάσεις ώστε να τροποποιήσει την ιεραρχία των κλάσεων, π.χ., να συμπεράνει ότι η Κλάση A είναι υποκλάση της Κλάσης B. **Anonymous** κλάσεις που δημιουργούνται αυτόματα από τον συμπεραστή όταν χρησιμοποιείουμε εκφράσεις κλάσεων.

Τέτοιους περιορισμούς στο Protégé μπορούμε να βάλουμε από το Class Description View που περιέχει το μεγαλύτερο μέρος των πληροφοριών που χρησιμοποιούνται για την περιγραφή μιας κλάσης. Βάζοντας τους περιορισμούς με χρήση του Subclass Of τότε λειτουργούν ως αναγκαίοι και η κλάση θα είναι primitive. Ενώ βάζοντας τους περιορισμούς με χρήση το Equivalent To λειτουργούν ως αναγκαίοι και επαρκείς και η κλάση γίνεται defined. Η περιγραφή και ο ορισμός των κλάσεων στην OWL αποτελέι μια από τις σημαντικότερες διαφορές σε σχέση με άλλα μοντέλα, όπως οι περισσότερες γλώσσες προγραμματισμού αντικειμενοστραφούς σχεδίασης.Καθώς με τους ορισμούς μπορούμε να εξηγούμε γιατί μια κλάση είναι υποκλάση μιας άλλης ενώ ο OWL ταξινομητής μπορεί να αναδομεί την ιεραρχία των κλάσεων.

Οι περιορισμοί που θα βάλουμε είναι για τις υποκλάσεις της Course ώστε ανάλογα με την τιμή της ιδιότητας schema/additionalType τα μαθήματα να ταξινομούνται στις αντίστοιχες κλάσεις. Αυτοί οι περιορισμοί θα οριστούν ως αναγκαίοι και επαρκείς άρα οι υποκλάσεις θα τεθούν ως Defined. Επιλέγουμε την CoCourse και απο το Description>Equivalent To>Class expression editor συμπληρώνουμε σε Manchester Syntax το ≪additionalType value ”Cumpulsory (CO)”≫. Για την OptionalCourse ακολουθούμε το ίδιο βάζοντας ≪additionalType value ”Optional (OP)”≫. Ενώ για τις ElesCourse και PdaiCourse με την ιδιότητα inGroup και “ELES”,”PDAI” αντίστοιχα.

Σχήμα 4.16: Περιορισμοί hasValule με Manchester syntax.

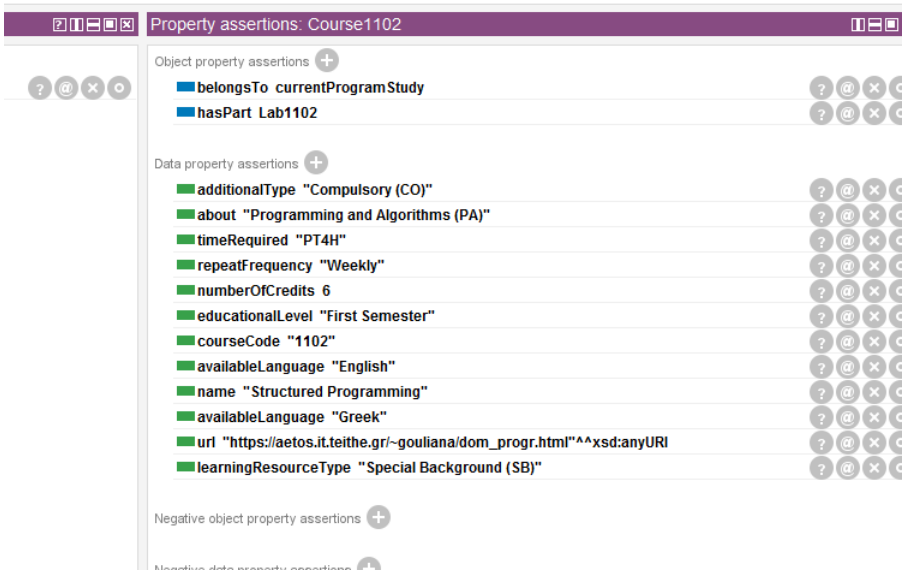
4.1.8 Δημιουργία ατόμων(Individuals)

Στο Protege απο την καρτέλα Individual by class έχουμε την δυνατότητα να προσθέσουμε άτομα στις κλάσεις μας. Ξεκινάμε προσθέτοντας το currentProgramStudy στην κλάση EducationalOccupationalProgram και απο την καρτέλα Property assertions θα εισάγουμε τα δεδομένα για το τρέχον

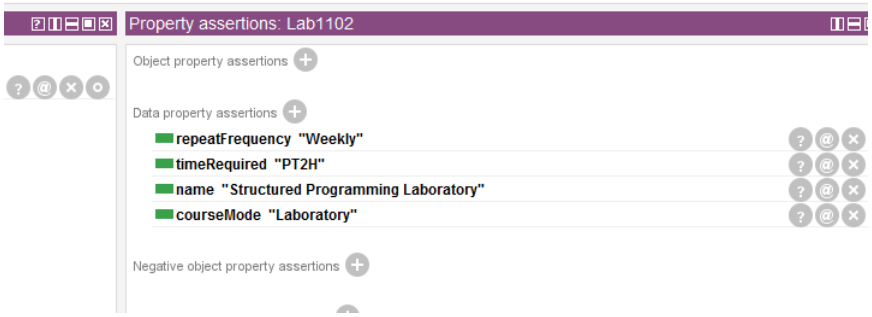
πρόγραμμα σπουδών.

Σχήμα 4.17: Τύποι δεδομένων Προγράμματος σπουδών.

Η εισαγωγή των μαθημάτων θα γίνει με τον ίδιο τρόπο αντιστοιχίζοντας τα δεδομένα από την διεύθυνση: https://www.iee.ihu.gr/udg_courses/. Τα εργαστηριακά μέρη θα περαστούν και αυτά στο μοντέλο αλλά με λιγότερα δεδομένα. Συγκεκριμένα οι τύποι δεδομένων για τα εργαστήρια θα ειναι το sch:courseMode για την δήλωση εργαστηρίου.Το sch:name για το όνομα του μαθήματος, sch:repeatFrequency για την συχνότητα που διδάσκεται και το sch:timeRequired για τις ώρες που απαιτούνται εβδομαδιαία. Ο συνολικός αριθμός των μαθήματων είναι 85 από τα οποία τα 28 έχουν και εργαστηριακό μερός. Αποθηκεύοντας την οντολογία το Protege παρέχει την δυνατότητα επιλογής δίαφορων μορφών όπως RDF/XML, Turtle, JSON-LD κ.α. Επιλέγοντας turtle και ανοίγοντας το αρχείο με κάποιον επεξεργαστή κειμένου που υποστηρίζει turtle σύνταξη μπορούμε να δούμε όλες της owl/rdf συσχετίσεις που έχουν δημιουργιθεί αλλά και να προσθέσουμε καινούριες.



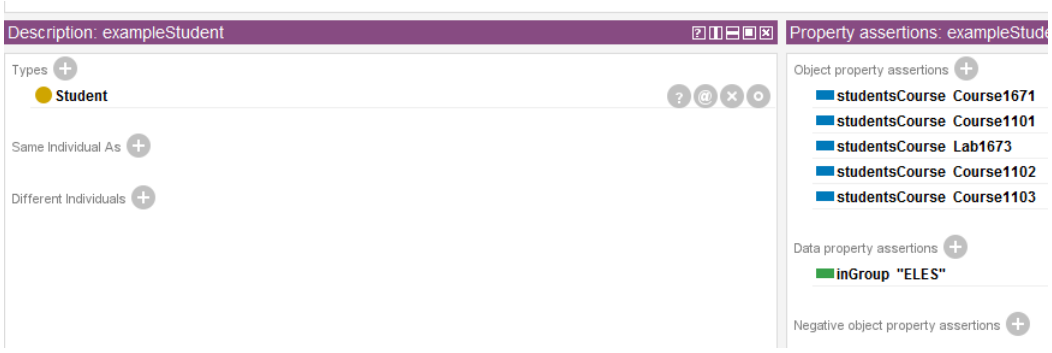
Σχήμα 4.18: Τύποι δεδομένων θεωριτικού μαθήματος.



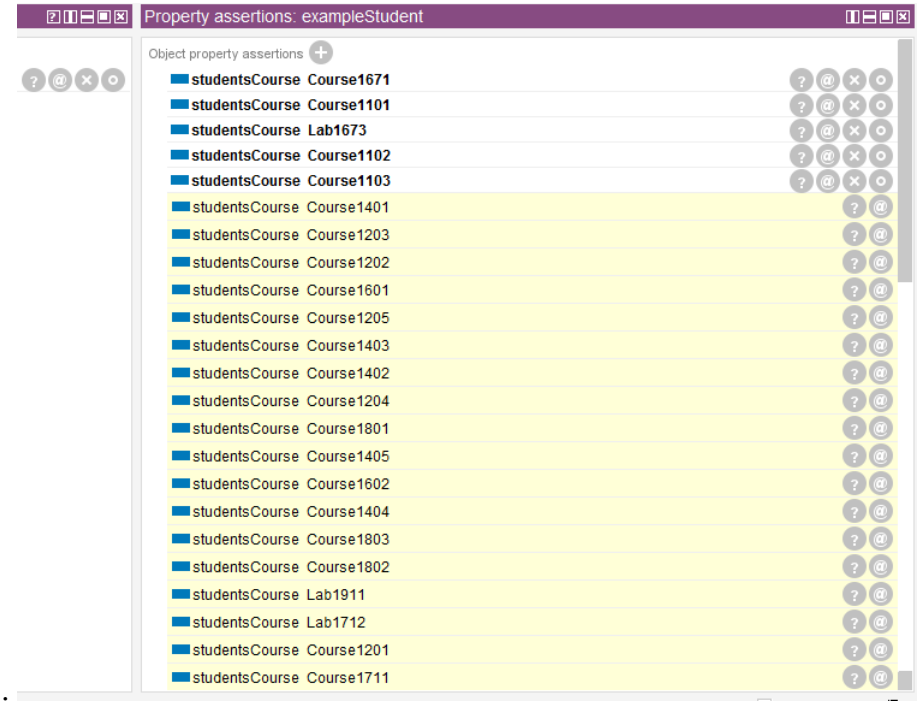
Σχήμα 4.19: Τύποι δεδομένων εργαστηριακού μαθήματος.

4.1.9 Δοκιμή μοντέλου

Έχοντας ολοκληρώσει την εισαγώγη όλων των μαθημάτων μπορούμε να ελέγξουμε και να επαληθεύσουμε το μοντέλο. Θα δημιουργούμε ένα άτομο της κλάσης Student με όνομα exampleStudent και θα δηλώσουμε τα μαθήματα που έχει περάσει είτε θεωρίας είτε εργαστήρια καθώς και την ομάδα στην οποία ανήκει. Τρέχοντας τον συμπεραστή το Protege θα μας εμφανίσει τις νέες συσχετίσεις του φοιτητή με τα κοινά υποχρεωτικά μαθήματα που δεν έχουν περαστεί και ομοίως για τα υποχρεωτικά της ομάδας του. Επίσης θα συσχετίσει των φοιτητή με το μέρος από μαθήματα επιλογής που χρωστάει ανάλογα αν έχει περάσει το θεωριτικό η το εργαστηριακό μέρος. Από την καρτέλα Individuals by class προσθέτουμε νέο άτομο και στην ιδιότητα αντικειμένου studentsCourse προσθέτουμε τα μαθήματα: Course1101, Course1102, Course1103, Course1671, Lab1673. Δηλώνουμε την ομάδα του με τον τύπο δεδομένων inGroup ως ”ELES” και ξεκινάμε τον συμπεραστή. Στο σχήμα 4.20 με κίτρινο χρώμα είναι τα μαθήματα που έχουν δημιουργιθεί με αυτόματο συμπερασμό τα οποία χρωστάει ο φοιτητής.



Σχήμα 4.20: Παράδειγμα φοιτητή στο μοντέλο.



Σχήμα 4.21: Δημιουργία νέων σχέσεων από τον συμπεραστή.

4.2 Συμπεράσματα και Περιορισμοί

Η προσέγγιση αυτή, που χρησιμοποιεί αυτόματη συμπερασματολογία για τον εντοπισμό των χρωστούμενων μαθημάτων (υποχρεωτικών ή επιλογής), δεν καλύπτει πλήρως τις απαιτήσεις του προβλήματος. Είναι σημαντικό να τονίσουμε ότι η OWL έχει σχεδιαστεί για την αναπαράσταση γνώσης στο διαδίκτυο και, για αυτόν τον λόγο, λειτουργεί υπό την υπόθεση του Ανοιχτού Κόσμου (OWA). Ωστόσο, η αρχή του Ανοιχτού Κόσμου δημιουργεί προβλήματα όταν προσπαθούμε να προσδιορίσουμε ποια μαθήματα δεν έχει περάσει ένας φοιτητής. Στην πράξη, δεν μπορούμε να ελέγξουμε εάν κάτι δεν υπάρχει, ώστε να εξαγάγουμε νέες συσχετίσεις μέσω συμπερασματολογίας. Επειδή η OWL υποθέτει ότι η γνώση είναι ανοιχτή και ενδέχεται να είναι ελλιπής, η απουσία μιας ιδιότητας (π.χ., studentsCourse) δεν σημαίνει ότι δεν ισχύει μπορεί απλώς να μην είναι γνωστή. Αυτό καθιστά αδύνατη την προσέγγιση μιας λύσης που βασίζεται στη δημιουργία νέων συσχετίσεων μέσω ελέγχου της μη ύπαρξης δεδομένων, όπως θα συνέβαινε με τη χρήση ενός τελεστή not.

Η υλοποιήση του μοντέλου δημιουργεί συσχετίσεις του φοιτητή με όλα τα κοινά υποχρεωτικά μαθήματα, τα υποχρεωτικά μαθήματα ομάδας, καθώς και με τα μαθήματα που έχει περάσει έστω και ένα από τα δύο μέρη (θεωρία ή εργαστήριο) για τα επιλογής. Αυτές οι συσχετίσεις γίνονται με την ίδια ιδιότητα αντικειμένου που χρησιμοποιούμε για τα ήδη περασμένα μαθήματα, δηλαδή την studentsCourse.Με τον τρόπο αυτό ο συμπεραστής δεν θα δημιουργίσει και-

νούριες συσχετίσεις αν υπάρχουν ήδη δηλωμένες.Βέβαια ως αποτέλεσμα, όλα τα μαθήματα είτε είναι περασμένα είτε χρωστούμενα, δηλώνονται μέσω της ίδιας ιδιότητας. Παρόλα αυτά, στο Protégé, μπορούμε να διακρίνουμε τη νέα γνώση που έχει προκύψει από τον συμπεραστή, καθώς εμφανίζεται με κίτρινο χρώμα, ξεχωρίζοντας από τις ήδη υπάρχουσες δηλώσεις μας.
Επιπλέον, οι περιορισμοί exactly και max δεν λειτουργούν όπως αναμένεται σε πολλές περιπτώσεις. Για παράδειγμα, αν θέλουμε να χαρακτηρίσουμε τους φοιτητές που έχουν περάσει ακριβώς 5 μαθήματα επιλογής ως φοιτητές που χρωστούν άλλα 3 δεν είναι εφικτό. Σύμφωνα με την OWA, μπορεί να υπάρχουν επιπλέον άγνωστα δεδομένα, επομένως η πληροφορία μας είναι ατελής. Παράλληλα, η OWL δεν υποστηρίζει συναθροιστικές συναρτήσεις, όπως η COUNT, για την αυτόματη εξαγωγή συμπερασμάτων, καθιστώντας αδύνατο τον υπολογισμό του πλήθους μαθημάτων που έχει περάσει ένας φοιτητής. Ενώ ακολουθεί και μονοτονικό συλλογισμό, πράγμα που σημαίνει ότι τα συμπεράσματα που εξάγονται μία φορά δεν μπορούν να αναιρεθούν από νέα δεδομένα. Αυτό αποτελεί πρόβλημα σε δυναμικά περιβάλλοντα όπου η γνώση αλλάζει συχνά, καθώς δεν μπορούμε να αφαιρέσουμε ή να αναθεωρήσουμε προηγούμενα συμπεράσματα με βάση νεότερες πληροφορίες.
Επομένως, για μια ολοκληρωμένη λύση, είναι απαραίτητη η χρήση μιας τεχνολογίας που λειτουργεί με την αρχή του Κλειστού Κόσμου (CWA), όπως η SPARQL, ώστε να ξεπεράσουμε όλους αυτούς τους περιορισμούς.

Κεφάλαιο 5

Διαχείριση Δεδομένων RDF

Εισαγωγή

Αν και το Protégé διαθέτει ενσωματωμένο επεξεργαστή ερωτημάτων SPARQL (SPARQL Editor), υπάρχει επίσης η δυνατότητα προσθήκης του ως επέκταση μέσω του Snap SPARQL. Ωστόσο, η χρήση και των δύο είναι αρκετά περιορισμένη, καθώς δεν υποστηρίζουν πλήρως όλες τις λειτουργίες της SPARQL. Ενδεικτικά, δεν υποστηρίζεται η εκτέλεση UPDATE ερωτημάτων για εισαγωγή και διαγραφή τριάδων. Η διαφορά μεταξύ του SPARQL Editor και του πρόσθετου Snap SPARQL είναι ότι το δεύτερο μπορεί να εκτελεί ερωτήματα πάνω στη γνώση που παράγεται από το μοντέλο μέσω του συμπεραστή. Επομένως, για να αξιοποιήσουμε πλήρως τις δυνατότητες της SPARQL, είναι απαραίτητη η χρήση ενός αποθετηρίου για την αποθήκευση και επεξεργασία OWL και RDF δεδομένων.

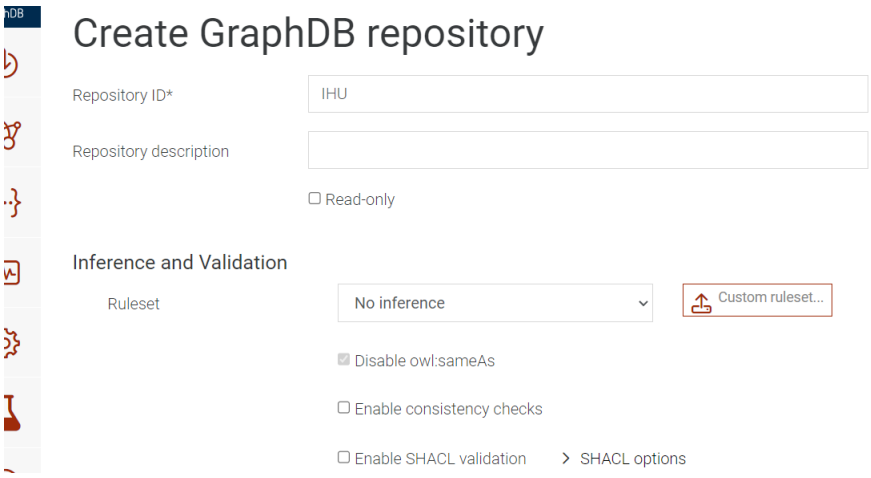
5.1 GraphDb

Το GraphDB είναι μια ισχυρή, επεκτάσιμη και αξιόπιστη βάση δεδομένων γραφημάτων, που αναπτύχθηκε από την Ontotext και εξειδικεύεται στις τεχνολογίες του Σημασιολογικού Ιστού και των Γραφημάτων Γνώσης. Η πρώτη έκδοσή του κυκλοφόρησε στις αρχές της δεκαετίας του 2000 με την ονομασία BigOWLIM. Το GraphDB παρέχει ευκολία χρήσης και πλήρη συμβατότητα με τα βιομηχανικά πρότυπα, εφαρμόζοντας τις διεπαφές του πλαισίου RDF4J, το πρωτόκολλο W3C SPARQL και υποστηρίζοντας όλες τις εκδόσεις RDF. Είναι η προτιμώμενη επιλογή τόσο για μεγάλες επιχειρήσεις όσο και για ανεξάρτητους προγραμματιστές, χάρη στη δυναμική κοινότητά του και την εμπορική υποστήριξη που προσφέρει η Ontotext. Ενσωματώνει μηχανές αναζήτησης, όπως Elasticsearch και Solr, επιτρέποντας γρήγορη και αποτελεσματική αναζήτηση δεδομένων. Παράλληλα, μπορεί να αποθηκεύσει και να επεξεργαστεί δισεκατομμύρια τριπλέτες RDF, είτε σε έναν απλό υπολογιστή είτε σε έναν εξυπηρετητή (server),

διατηρώντας υψηλή απόδοση. Επιπλέον, το GraphDB μπορεί να αναπτυχθεί σε cluster (σύνολο διακομιστών), όπου τα δεδομένα και οι ερωτήσεις SPARQL κατανέμονται σε πολλαπλές μηχανές, αυξάνοντας την ταχύτητα και την αξιοπιστία του συστήματος. Είναι από τα ελάχιστα triplestores που υποστηρίζουν λογικό συμπερασμό (reasoning) σε μεγάλη κλίμακα, επιτρέποντας την εξαγωγή νέας γνώσης από υπάρχουσες πληροφορίες. Το GraphDB είναι διαθέσιμο και ως υπηρεσία cloud σε Amazon Web Services (AWS) και Microsoft Azure. Η Ontotext προσφέρει δύο εκδόσεις την GraphDB Free, που διατίθεται δωρεάν και την GraphDB Enterprise (EE), η οποία υποστηρίζει μεγάλης κλίμακας ερωτήματα που απαιτούν αυξημένους υπολογιστικούς πόρους.

5.1.1 Εγκατασταση

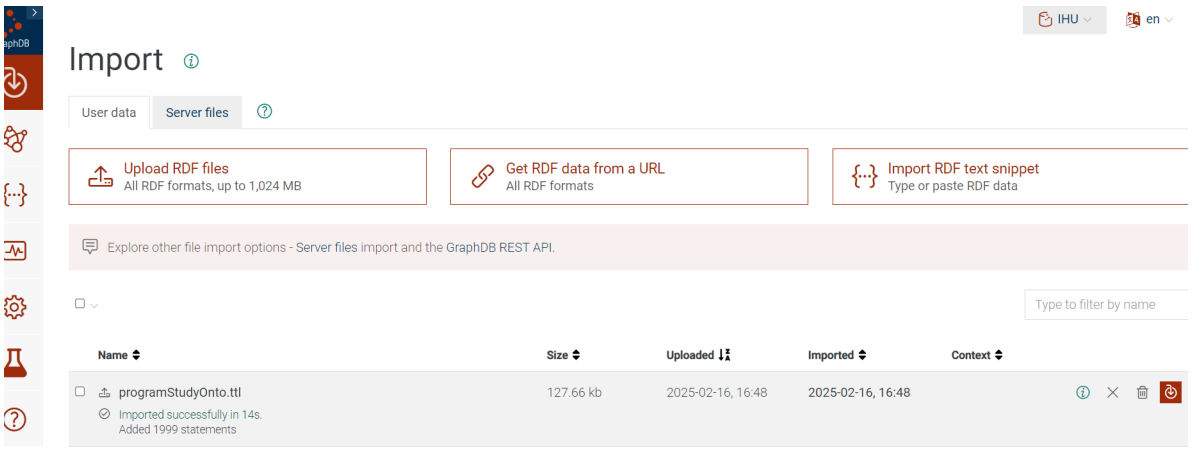
Αρχικά, μεταβαίνουμε στη σελίδα λήψης του GraphDB (https://www.ontotext.com/products/graphdb/), συμπληρώνουμε τα στοιχεία μας και ζητάμε ένα αντίγραφο. Στο email που δηλώσαμε, θα λάβουμε έναν σύνδεσμο λήψης και, ανάλογα με το λειτουργικό μας σύστημα, ακολουθούμε τις αντίστοιχες οδηγίες. Η εγκατάσταση πραγματοποιείται μέσω γραφικού περιβάλλοντος χρήστη (GUI), χωρίς να απαιτείται η χρήση κονσόλας. Επιπλέον, δεν είναι απαραίτητο να κατεβάσουμε ξεχωριστά τη Java, καθώς περιλαμβάνεται μαζί με το GraphDB. [20]
Με την ολοκλήρωση της εγκατάστασης, το περιβάλλον του GraphDB θα ανοίξει αυτόματα στον προεπιλεγμένο φυλλομετρητή, στη διεύθυνση http://localhost:7200/. Υπάρχει η δυνατότητα δημιουργίας διαφορετικών προφίλ χρηστών και διαχειριστών, ανάλογα με τα δικαιώματα που θέλουμε να παραχωρήσουμε στον καθένα. Από την επιλογή Setup, μπορούμε να δημιουργήσουμε ένα νέο αποθετήριο (repository) με το όνομα IHU. Δεδομένου ότι τα ερωτήματα που θα εκτελέσουμε δεν απαιτούν συμπερασμό (inferencing), επιλέγουμε την επιλογή No inference.



Σχήμα 5.1: Δημιουργία αποθετηρίου με όνομα IHU.

Στη συνέχεια, από την καρτέλα Import, επιλέγουμε το αποθετήριο και φορτώνουμε την οντο-

λογία που έχουμε δημιουργήσει, σε αρχείο τύπου Turtle.



Σχήμα 5.2: Φόρτωση του αρχείου με την οντολογία στο αποθετήριο.

5.1.2 Λειτουργίες Explore

- Graphs overview: Γίνεται επισκόπηση του γραφήματος σε μορφή πίνακα με όλα τα στοιχεία που έχουμε περάσει στην οντολογία. Από εδώ μπορούμε να πάρουμε την οντολογία σε έναν αρχείο σαν αυτό που φορτώσαμε ή και σε άλλη μορφή όπως JSON, RDF-XML, N-triples.

- Class hierarchy: Μπορούμε να δούμε σε ένα διάγραμμα την απεικόνιση της ιεραρχίας των κλάσεων που περιέχει η οντολογία μας από τον αριθμό των στιγμιοτύπων (instances). Οι μεγαλύτεροι κύκλοι είναι οι γονικές κλάσεις (parent classes) και οι εμφωλευμένοι κύκλοι είναι τα παιδιά τους.

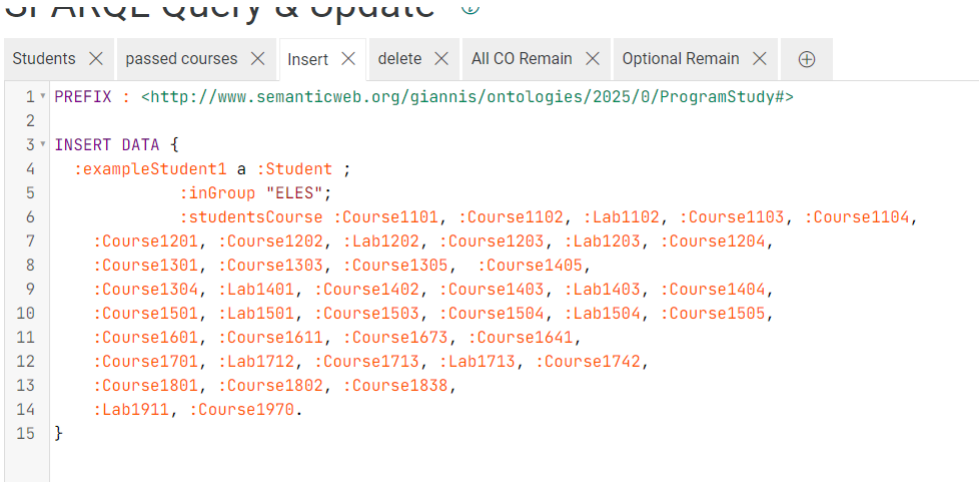
- Class relationships: Μπορούμε να δούμε ένα περίπλοκο διάγραμμα που παρουσιάζει μόνο τις κύριες σχέσεις, όπου κάθε μια από αυτές είναι μια δέσμη συνδέσεων μεταξύ των ξεχωριστών περιπτώσεων δύο κλάσεων. Κάθε σύνδεση είναι μια δήλωση RDF όπου το υποκείμενο είναι στιγμιότυπο μιας κλάσης, το αντικείμενο είναι μια στιγμιότυπο μιας άλλης κλάσης, και η σύνδεση είναι το κατηγόρημα. Ανάλογα με τον αριθμό συνδέσεων μεταξύ των στιγμιότυπων από δύο κλάσεις, η δέσμη μπορεί να είναι παχύτερη ή λεπτότερη και παίρνει το χρώμα της κλάσης με τις περισσότερες εισερχόμενες συνδέσεις. Αυτές οι συνδέσεις μπορεί να είναι και στις δύο κατευθύνσεις.

- Visual graph: Βλέπουμε το γράφημα της οντολογίας και μπορούμε να περιηγηθούμε σε αυτό χωρίς να κάνουμε χρήση ερωτημάτων SPARQL. Υπάρχει πεδίο αναζήτησης για να εισάγουμε τον όρο από τον οποίο θα ξεκινάει η εξερεύνηση του γράφου μας.

- Similarity: Επιτρέπει τον εντοπισμό ομοιοτήτων μεταξύ διαφορετικών οντοτήτων, βασισμένο σε κριτήρια όπως κοινά χαρακτηριστικά ή σχέσεις.

5.1.3 Sparql ερωτήματα έυρεσης χρωστούμενων μαθημάτων

Ανοίγοντας τον επεξεργαστή SPARQL, θα ξεκινήσουμε προσθέτοντας έναν φοιτητή με την ομάδα την οποία έχει δηλώσει και τα μαθήματα που έχει περάσει.



Σχήμα 5.3: Εισαγωγή φοιτητή με update query.

Στην γραμμή 4 δηλώνουμε το άτομο (individual) exampleStudent1 και, με τη λέξη-κλειδί a (συντομογραφία του rdf:type), ορίζουμε ότι ανήκει στην κλάση Student. Στην γραμμή 5, με την ιδιότητα inGroup, δηλώνουμε την ομάδα του, ενώ στην γραμμή 6, με την ιδιότητα studentCourse, καταγράφουμε όλα τα μαθήματα που έχει περάσει. Αυτό που θέλουμε είναι να επιστραφούν όλα τα κοινά υποχρεωτικά μαθήματα που δεν έχουν περαστεί, όλα τα υποχρεωτικά μαθήματα ομάδας που δεν έχουν περαστεί και σε περίπτωση που ένα μάθημα επιλογής έχει περαστεί μερικώς, να επιστραφεί το μέρος που απομένει. Και τα τρία αυτά ζητούμενα μπορούμε να τα συμπεριλάβουμε σε ένα ενιαίο SPARQL ερώτημα, συνδέοντάς τα με UNION. Για το πρώτο κομμάτι, ξεκινάμε επιλέγοντας τα μαθήματα όπου η ιδιότητα additionalType είναι ”Compulsory (CO)”. Στη συνέχεια, φιλτράρουμε τα αποτελέσματα, κρατώντας μόνο εκείνα που δεν συνδέονται με την ιδιότητα αντικειμένου studentCourse με τον φοιτητή exampleStudent1. Έπειτα, με τον ίδιο τρόπο, επιλέγουμε τα εργαστήρια που ανήκουν σε κάποιο υποχρεωτικό μάθημα αλλά δεν έχουν περαστεί από τον φοιτητή. Για το δεύτερο κομμάτι, ακολουθούμε την ίδια διαδικασία, με την προσθήκη της ιδιότητας inGroup, ώστε να ελέγξουμε ότι τα μαθήματα ανήκουν στην ίδια ομάδα με τον φοιτητή.

Students | passed courses | Insert | delete | All CO Remain | Optional Remain

```
PREFIX sch: <http://www.schema.org/>
PREFIX : <http://www.semanticweb.org/giannis/ontologies/2025/0/ProgramStudy#>

SELECT DISTINCT ?Course ?category
WHERE {
  {
    # CO unpassed
    ?Course a sch:Course ;
            sch:additionalType "Compulsory (CO)" .
    FILTER NOT EXISTS { :exampleStudent1 :studentsCourse ?Course . }
    BIND("CO unpassed" AS ?category)
  }
  UNION
  {
    # CO unpassed labs
    ?course a sch:Course ;
            sch:additionalType "Compulsory (CO)" ;
            sch:hasPart ?Course .
    FILTER NOT EXISTS { :exampleStudent1 :studentsCourse ?Course . }
    BIND("CO unpassed" AS ?category)
  }
```

Σχήμα 5.4: Κοινά υποχρεωτικά μαθήματα που δεν έχουν περαστεί.

```
    UNION
  {
    # GroupCO unpassed
    ?Course a sch:Course ;
            sch:additionalType "Compulsory Optional (CO-OP)" ;
            :inGroup ?group .
    :exampleStudent1 :inGroup ?group .
    FILTER NOT EXISTS { :exampleStudent1 :studentsCourse ?Course . }
    BIND("GroupCO unpassed" AS ?category)
  }
  UNION
  {
    # GroupCO unpassed labs
    ?course a sch:Course ;
            sch:additionalType "Compulsory Optional (CO-OP)" ;
            :inGroup ?group ;
            sch:hasPart ?Course .
    :exampleStudent1 :inGroup ?group .
    FILTER NOT EXISTS { :exampleStudent1 :studentsCourse ?Course . }
    BIND("GroupCO unpassed" AS ?category)
  }

```

Σχήμα 5.5: Υποχρεωτικά μαθήματα ομάδας που δεν έχουν περαστεί.

Ενώ για το τρίτο μέρος ανάλογα την ομάδα του φοιτητή βρίσκουμε τα μαθήματα επιλογής και φιλτράρουμε όσα έχουν περαστέι μισά, είτε θεωρία είτε εργαστήριο.

```
    UNION
  {
    # Remaining Optional parts
    ?course a sch:Course ;
            sch:additionalType ?type ;
            sch:hasPart ?lab .
    VALUES ?type { "Optional (OP)" "Compulsory Optional (CO-OP)" }

    # Match Optional (OP) or CO-OP in other groups
    OPTIONAL { ?course :inGroup ?courseGroup . }
    :exampleStudent1 :inGroup ?studentGroup .
   FILTER (!BOUND(?courseGroup) || ?courseGroup != ?studentGroup || ((?type="Optional (OP)")&&(?courseGroup = ?studentGroup)))
    # Remaining parts logic
    FILTER (
      (EXISTS { :exampleStudent1 :studentsCourse ?course } && NOT EXISTS { :exampleStudent1 :studentsCourse ?lab }) ||
      (EXISTS { :exampleStudent1 :studentsCourse ?lab } && NOT EXISTS { :exampleStudent1 :studentsCourse ?course })
    )
    BIND(IF(EXISTS { :exampleStudent1 :studentsCourse ?course }, ?lab, ?course) AS ?Course)
    BIND("Optional unpassed" AS ?category)
  }

}
```

Σχήμα 5.6: Μέρη από μαθήματα επιλογής που δεν έχουν περαστεί.

	Course	category
1	:Course1105	"CO unpassed"
2	:Course1205	"CO unpassed"
3	:Course1302	"CO unpassed"
4	:Course1401	"CO unpassed"
5	:Course1502	"CO unpassed"
6	:Course1602	"CO unpassed"
7	:Course1702	"CO unpassed"
8	:Course1803	"CO unpassed"
9	:Course1999	"CO unpassed"
10	:Lab1204	"CO unpassed"
11	:Lab1205	"CO unpassed"
12	:Lab1304	"CO unpassed"
13	:Lab1405	"CO unpassed"
14	:Lab1602	"CO unpassed"
15	:Lab1701	"CO unpassed"
16	:Course1711	"GroupCO unpassed"
17	:Course1712	"GroupCO unpassed"
18	:Course1911	"GroupCO unpassed"
19	:Course1912	"GroupCO unpassed"
20	:Lab1611	"GroupCO unpassed"
21	:Lab1673	"Optional unpassed"
22	:Lab1838	"Optional unpassed"

Σχήμα 5.7: Αποτελέσματα ερωτήματος.

5.1.4 Sparql ερωτήματα έυρεσης μαθημάτων επιλογής

Για κάθε φοιτητή, χρειάζεται επίσης να εμφανίσουμε τον αριθμό των εναπομεινάντων μαθημάτων επιλογής, σύμφωνα με τις πιστωτικές μονάδες που έχει συγκεντρώσει από τα επιτυχώς περασμένα μαθήματα. Έτσι, θα μπορούμε να υπολογίσουμε σωστά τον αριθμό των χρωστούμενων μαθημάτων επιλογής, αφού κάθε μάθημα αντιστοιχεί σε 6 πιστωτικές μονάδες, ενώ η πρακτική άσκηση σε 12. Με το παρακάτω ερώτημα, θέλουμε να επιλέξουμε για έναν φοιτητή τα περασμένα μαθήματα επιλογής, τον αριθμό αυτών, τις πιστωτικές μονάδες που του αντιστοιχούν και τον αριθμό των χρωστούμενων μαθημάτων επιλογής. Ξεκινάμε με τα μαθήματα που έχουν την ιδιότητα additionalType ίση με ”Optional (OP)” και προσθέτουμε τις πιστωτικές μονάδες μόνο αν ο φοιτητής έχει περάσει το μάθημα και, αν υπάρχει, το εργαστήριο.

```
PREFIX sch: <http://www.schema.org/>
PREFIX : <http://www.semanticweb.org/giannis/ontologies/2025/0/ProgramStudy#>

SELECT (GROUP_CONCAT(REPLACE(STR(?course), ".*#", ""); separator=", ") AS ?passedCourses)
       (COUNT(DISTINCT ?course) AS ?passedOptionalCount)
        (sum(?credits) as ?TotalCredits)
        ((48 - sum(?credits)) /6 AS ?remainingOptionalCourses)
WHERE {
  {
    # Match courses with type "Optional (OP)"
    ?course a sch:Course ;
            sch:additionalType "Optional (OP)" ;
            sch:numberOfCredits ?credits .

    # Check if the course has a lab
    OPTIONAL {
      ?course sch:hasPart ?lab .
    }

    # Ensure the course is passed
    FILTER EXISTS {
      :exampleStudent1 :studentsCourse ?course .
    }

    # If the course has a lab, ensure the lab is also passed
    FILTER (!BOUND(?lab) || EXISTS {
      :exampleStudent1 :studentsCourse ?lab .
    })
  }
```

Σχήμα 5.8: Πιστωτικές μονάδες από μαθήματα επιλογής.

Με τον ίδιο τρόπο, υπολογίζουμε και προσθέτουμε τις πιστωτικές μονάδες για τα μαθήματα με την ιδιότητα additionalType ίση με ”Compulsory Optional (CO-OP)”, αλλά μόνο όταν αυτά ανήκουν σε διαφορετική ομάδα από αυτήν του φοιτητή. Τέλος, τα αποτελέσματα του ερωτήματος φαίνονται στο Σχήμα 5.10.

```
30  UNION
31  {
32    # Match courses with type "Compulsory Optional (CO-OP)"
33    ?course a sch:Course ;
34            sch:additionalType "Compulsory Optional (CO-OP)" ;
35            :inGroup ?courseGroup ;
36            sch:numberOfCredits ?credits .
37
38    # Match the student's chosen study group
39    :exampleStudent1 :inGroup ?studentGroup .
40
41    # Include only courses not in the same group as the student's choice group
42    FILTER (?courseGroup != ?studentGroup)
43
44    # Check if the course has a lab
45    OPTIONAL {
46      ?course sch:hasPart ?lab .
47    }
48
49    # Ensure the course is passed
50    FILTER EXISTS {
51      :exampleStudent1 :studentsCourse ?course .
52    }
53
54    # If the course has a lab, ensure the lab is also passed
55    FILTER (!BOUND(?lab) || EXISTS {
56      :exampleStudent1 :studentsCourse ?lab .
57    })
58    }}
```

Σχήμα 5.9: Πιστωτικές μονάδες από μαθήματα επιλογής διαφορετικής ομάδας.

	passedCourses	passedOptionalCount	TotalCredits	remainingOptionalCourses
1	"Course1713, Course1970, Course1641, Course1742"	"4"^^xsd:integer	"30"^^xsd:integer	"3"^^xsd:decimal

Σχήμα 5.10: Αποτελέσματα ερωτήματος.

5.2 Apache Jena Cli

Πρόκειται για ένα λογισμικό ανοιχτού κώδικα βασισμένο στη Java για την ανάπτυξη εφαρμογών Σημασιολογικού Ιστού και Συνδεδεμένων Δεδομένων(Linked Data). Υποστηρίζει ολες τις βασικές τεχνολογίες παρέχοντας εργαλεία για τη δημιουργία, διαχείριση και ανάκτηση δεδομένων RDF. Περιλαμβάνει μια μηχανή λογικού συλλογισμού βασισμένη σε κανόνες, καθώς και το TDB, μια επεκτάσιμη βάση αποθήκευσης τριπλετών RDF. Επιπλέον, διαθέτει τον εξυπηρετητή Fuseki, ο οποίος επιτρέπει την απομακρυσμένη εκτέλεση ερωτημάτων SPARQL πάνω σε RDF δεδομένα. Το Jena χρησιμοποιείται ευρέως σε εφαρμογές που σχετίζονται με σημασιολογικά δεδομένα και αναπαράσταση γνώσης. [21]

Αρχικά Το Jena 5 απαιτεί να έχουμε εγκατεστημένη την έκδοση Java 17 ή νεότερη. Κατεβάζουμε τις απαραίτητες βιβλιοθήκες από τη διεύθυνση https://jena.apache.org/download/index.cgi. Αφού αποσυμπιέσουμε το αρχείο, δημιουργούμε μια μεταβλητή περιβάλλοντος χρήστη JENA_HOME,

θέτοντας ως τιμή τη διαδρομή του φακέλου εγκατάστασης του Jena. Στη συνέχεια, προσθέτουμε στη μεταβλητή περιβάλλοντος του συστήματος Path τη διαδρομή προς τον φάκελο bat, ώστε να μπορούμε να χρησιμοποιούμε τις εντολές του Jena από τη γραμμή εντολών. Ανοίγοντας τη γραμμή εντολών, ελέγχουμε αν η εγκατάσταση έχει γίνει σωστά με την εντολή: arq –version

```
C:\Users\User\Desktop>arq --version
Apache Jena version 5.4.0
```

Σχήμα 5.11: Ελεγχος εγκατάστασης.

Στη συνέχεια, έχοντας την οντολογία με όνομα αρχείου onto.ttl εκτελούμε ένα ερώτημα για την εμφάνιση όλων των συσχετίσεων για το μάθημα με κωδικό ”1403” (Σχήμα 5.12). Αποθηκεύουμε το ερώτημα σε αρχείο με όνομα qr.txt και το εκτελούμε με την εντολή: arq –data onto.ttl –query qr.txt (Σχήμα 5.13).

```
1 PREFIX sch: <http://www.schema.org/>
2 PREFIX : <http://www.semanticweb.org/giannis/ontologies/2025/0/ProgramStudy#>
3
4 SELECT ?p ?o
5 where{ :Course1403 ?p ?o}
```

Σχήμα 5.12: Αναζήτηση συσχετίσεων για το μάθημα με κωδικό ”1403”.

```
C:\Users\User\Desktop>arq --data onto.ttl --query qr.txt
-----------------------------------------------------------------------------------------------------------------------------------------------
| p                                                                | o                                                                        |
===============================================================================================================================================
| sch:additionalType                                               | "Compulsory (CO)"                                                        |
| <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>                | <http://www.w3.org/2002/07/owl#NamedIndividual>                          |
| <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>                | sch:Course                                                               |
| sch:coursePrerequisites                                          | :Course1103                                                              |
| sch:coursePrerequisites                                          | :Course1102                                                              |
| sch:educationalLevel                                             | "Fourth Semester"                                                        |
| sch:repeatFrequency                                              | "Weekly"                                                                 |
| :belongsTo                                                       | :currentProgramStudy                                                     |
| sch:url                                                          | "https://1403.iee.ihu.gr/"^^<http://www.w3.org/2001/XMLSchema#anyURI>    |
| sch:courseCode                                                   | "1403"                                                                   |
| sch:hasPart                                                      | :Lab1403                                                                 |
| sch:availableLanguage                                            | "Greek"                                                                  |
| sch:numberOfCredits                                              | 6                                                                        |
| sch:learningResourceType                                         | "Scientific Area - Skills Development (SA-SD)"                           |
| sch:timeRequired                                                 | "PT4H"                                                                   |
| sch:about                                                        | "Embedded Computation Systems (ECS)"                                     |
| sch:name                                                         | "Introduction to Operating Systems"                                      |
-----------------------------------------------------------------------------------------------------------------------------------------------

C:\Users\User\Desktop>
```

Σχήμα 5.13: Αποτελέσματα ερωτήματος.

Βιβλιογραφία

[1] T. Berners-Lee, J. Hendler και O. Lassila, "The Semantic Web", *Scientific American*, 2001.

[2] P. Hitzler, M. Krötzsch και S. Rudolph, *Foundations of Semantic Web Technologies*. Chapman & Hall/CRC, 2009, isbn: 978-1-4200-9050-5.

[3] G. Antoniou, P. Groth, F. van Harmelen και R. Hoekstra, *A Semantic Web Primer*, 3η έκδοση. Cambridge, MA: MIT Press, 2012, isbn: 9780262018289.

[4] C. M. Keet, *An Introduction to Ontology Engineering*, v1.5. 2020.

[5] G. Wu, επιμελητής, *Semantic Web*. Rijeka, Croatia: IntechOpen, 2010, isbn: 978-953-51-5745-8. διεύθν.: https://www.intechopen.com/books/3747.

[6] Σ. Γεώργιος, *Αναπαράσταση Οντολογικής Γνώσης και Συλλογιστική*. Κάλλιπος, Ανοικτές Ακαδημαϊκές Εκδόσεις, 2015, Διαθέσιμο στο: https://repository.kallipos.gr/handle/11419/4225, isbn: 978-960-603-157-1.

[7] N. D. Rodríguez, *Lecture 1: Introduction to Description Logics and OWL*, https://perso.telecom-paristech.fr/bloch/OptionIA/IA301-Lecture1-IntroDL-OWL.pdf, 2023.

[8] A. Artasanchez, *Creating Ontologies and Vocabularies to Support Semantic Search*, http://thedatascience.ninja/2023/06/22/creating-ontologies-and-vocabularies-to-support-semantic-search/, 2023.

[9] Σ. Μιχαήλ, Π. Ιωάννης και Α. Θεόδωρος, *Ανοικτά συνδεδεμένα δεδομένα και εφαρμογές: Μια πρακτική προσέγγιση στον σημασιολογικό ιστό*. Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών, 2015, Διαθέσιμο στο: https://repository.kallipos.gr/handle/11419/1338, isbn: 978-960-603-393-3.

[10] J. E. L. Gayo, E. Prud'hommeaux, I. Boneva, D. Kontokostas, Y. Ding και P. Groth, *Validating RDF Data* (Synthesis Lectures on the Semantic Web: Theory and Technology). Morgan & Claypool Publishers, 2017, isbn: 9781681731643.

[11] W3C, *RDF 1.1 Primer*, https://www.w3.org/TR/rdf11-primer/, 2014.

[12] W. W. W. C. (W3C), *Data on the Web Best Practices*, https://www.w3.org/TR/dwbp/, 2015.

[13] J. Hebeler, M. Fisher, R. Blace και A. Perez-Lopez, *Semantic Web Programming*. Wiley Publishing, Inc, 2009, isbn: 978-0-470-41801-7.

[14] H. Knublauch και D. Kontokostas, *Shapes Constraint Language (SHACL)*, https://www.w3.org/TR/shacl/, 2017.

[15] M. DeBellis, *Protégé 5 New OWL Pizza Tutorial V3.2*, https://www.michaeldebellis.com/post/new-protege-pizza-tutorial, 2021.

[16] D. Bob, *Learning SPARQL: Querying and updating with SPARQL 1.1*, 2nd. Sebastopol, CA: O'Reilly Media, 2013, isbn: 9781449371432.

[17] *Protégé - Documentation and Support*, https://protege.stanford.edu/support.php\#documentationSupport, Stanford Center for Biomedical Informatics Research.

[18] M. DeBellis, *The People Example Ontology V1*, https://drive.google.com/file/d/1QgDoHuFwgLOj2_EhpJTpQA9vTAhF7-IP/view, 2021.

[19] Schema.org, *Schema.org Documentation: Documents*, https://schema.org/docs/documents.html.

[20] Ontotext, *GraphDB Documentation*, https://graphdb.ontotext.com/documentation/11.0/.

[21] Apache Jena, *Apache Jena Documentation*, https://jena.apache.org/documentation/index.html, 2024.