ΑΝΑΠΤΥΞΗ ΟΝΤΟΛΟΓΙΑΣ ΣΤΗΝ ΚΑΡΔΙΟΛΟΓΙΑ ΓΙΑ ΑΝΑΚΤΗΣΗ ΚΕΙΜΕΝΩΝ

Η
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

Υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης
του Τμήματος Πληροφορικής
Εξεταστική Επιτροπή

από τον

Γεώργιο Λίτσιο

ως μέρος των Υποχρεώσεων

για τη λήψη

του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ ΣΤΙΣ ΤΕΧΝΟΛΟΓΙΕΣ-ΕΦΑΡΜΟΓΕΣ

Φεβρουάριος 2009

ΑΦΙΕΡΩΣΗ

Στην οικογένεια μου και τους φίλους μου που με στήριξαν στην όλη προσπάθειά μου για την ολοκλήρωση του μεταπτυχιακού μου.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου Δημήτριο Φωτιάδη για την ευκαιρία που μου έδωσε να γνωρίσω ένα τόσο νέο αντικείμενο στο πεδίο της πληροφορικής. Ένα μεγάλο ευχαριστώ επίσης στην οικογένεια μου που με στήριξε τόσο χρηματικά όσο και ψυχολογικά καθ'όλη τη διάρκεια των σπουδών μου. Δεν θα μπορούσα να ξεχάσω την αμέριστη συμπαράσταση των φίλων και συναδέλφων μου, που ήταν πάντα κοντά μου και μου δίνανε δύναμη να συνεχίσω την προσπάθειά μου καθημερινά (ιδίως της Μαρίας Κολτσίδα που ήταν κοντά μου σε στιγμές μεγάλου άγχους και μου θύμιζε καθημερινά το πόσο κοντά στην επίτευξη του στόχου ήμουν, καθώς και της Μαρίας Τζίμα και του Γεώργιου Γκανιάτσα που εκτός της συμπαράστασής τους με βοήθησαν και σε θέματα που είχαν να κάνουν με τη μεταπτυχιακή εργασία). Θα ήθελα επίσης, να ευχαριστήσω τον Δημήτριο Γάτσιο που μου έδωσε τη γνώση που είχε στο πεδίο των οντολογιών καθώς και τον Ευστράτιο Κοντόπουλο που με καθοδήγησε σε τεχνικά θέματα χρήσης οντολογιών σε εφαρμογές. Τέλος να πω ένα μεγάλο ευχαριστώ στον ιατρό Άρη Μπεχλιούλη που αφιέρωσε πολύτιμο χρόνο για να κατασκευάσουμε την οντολογία που ήταν μέρος της μεταπτυχιακής εργασίας.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΠΙΝΑΚΩΝ

ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

ΠΕΡΙΛΗΨΗ

Γεώργιος Λίτσιος του Θεοδώρου και της Δήμητρας. MSc, Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Φεβρουάριος, 2009. Ανάπτυξη Οντολογίας στην Καρδιολογία για Ανάκτηση Κειμένων. Επιβλέποντας: Δημήτριος Φωτιάδης.

Σκοπός της μεταπτυχιακής εργασίας είναι η μελέτη και δημιουργία οντολογίας στο πεδίο της καρδιολογίας, και πιο συγκεκριμένα στο πεδίο των καρδιαγγειακών νοσημάτων, και η μετέπειτα χρήση της σε ένα σύστημα ανάκτησης κειμένων. Οι οντολογίες θεωρούνται απαραίτητες για την μοντελοποίηση ενός πεδίου ενδιαφέροντος, καθώς και για την οργάνωση και τον διαμοιρασμό της γνώσης και πληροφορίας που υπάρχει διαθέσιμη. Ιδιαίτερη ανάγκη για οντολογίες υπάρχει στη σχεδίαση και υλοποίηση συστημάτων πληροφορίας στον τομέα της ιατρικής. Στην εργασία γίνεται θεωρητική ανασκόπηση των οντολογιών και μελετώνται τεχνολογίες για την ανάπτυξη και υλοποίησή τους. Υλοποιείται οντολογία με χρήση του εργαλείου ανάπτυξης οντολογιών Protégé και του UMLS. Επιπλέον σχεδιάζεται και υλοποιείται ένα σύστημα που κάνει χρήση της οντολογίας, για την ανάκτηση εγγράφων. Υποδεικνύεται έτσι ο τρόπος χρήσης των οντολογιών και η ενσωμάτωσή τους σε πραγματικά συστήματα που χρησιμοποιούνται στην κλινική πράξη.

EXTENDED ABSTRACT IN ENGLISH

Georgios Th. Litsios. M.Sc, Computer Science Department, University of Ioannina, February 2009. Development of Ontology in the Field of Cardiology and its Use in an Ontology-Based Document Retrieval System. Supervisor: Dimitrios I. Fotiadis.

The purpose of this study is the research, design and implementation of an Ontology in the field of cardiology, and more specifically in the cardiovascular diseases domain, and its utilization in an Ontology-based document retrieval system. Ontologies are considered to be of great importance while trying to model a specific field of interest as well as for the organisation, sharing and retrieval of available but in a scattered way knowledge. It appears that especially in the medical field, there is an excessive need for ontologies when designing and implementing computer information systems.

This study has been realized in three phases. In the first phase the theoretical background of ontologies is described. At the same time the state of art concerning technological background of implementing ontologies is analyzed. During the second phase, the ontology structure for cardiovascular diseases is built using well known tools and lexicons. (Protégé και UMLS). The third phase is related to the design and implementation of the ontology-based document retrieval computer system which uses the above described ontology. This study demonstrates the employment of ontologies and their use in real systems which are used in everyday clinical practice.

ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

1.1. Στόχοι της Διατριβής

Σήμερα οι οντολογίες παίζουν σημαντικό ρόλο στους τομείς της μηχανικής της γνώσης (knowledge engineering), της τεχνητής νοημοσύνης (artificial intelligence) και της πληροφορικής, σε πολλά πρακτικά πεδία, όπως στη μετάφραση της φυσικής γλώσσας, το ηλεκτρονικό εμπόριο, σε συστήματα γεωγραφικών πληροφοριών, σε συστήματα νομικών πληροφοριών, στην βιολογία και την ιατρική. Οι οντολογίες αποτελούν τη ραχοκοκαλιά αξιόπιστων και αποτελεσματικών εφαρμογών στον τομέα της Ιατρικής Φροντίδας και μπορούν να βοηθήσουν στην ανάπτυξη πιο ισχυρών και πιο διαλειτουργικών συστημάτων ιατρικής πληροφορίας. Η σχεδίαση και υλοποίηση των οντολογιών, στον τομέα της ιατρικής, εστιάζει κυρίως στην αναδιοργάνωση της ιατρικής ορολογίας [1].

Οι οντολογίες είναι το κέντρο των συζητήσεων σήμερα στην κοινότητα της ιατρικής πληροφορικής (medical informatics). Ο κυρίαρχος ρόλος τους στη σχεδίαση και υλοποίηση συστημάτων πληροφορίας στον τομέα της ιατρικής φροντίδας, σήμερα επιβεβαιώνεται ευρέως. Στη δύσκολη διαδικασία που μέχρι και τώρα είναι ανεπαρκής, του συντονισμού μεταξύ της παλιάς και νέας φυσικής γλώσσας που περιγράφει την ιατρική γνώση με όρους κοινά αποδεκτούς από τους εμπλεκόμενους στο πεδίο αυτό, καθώς και των μεθόδων, εργαλείων και συστημάτων πληροφορικής, η Ιατρική Οντολογία είναι μια από τις λίγες ελπίδες να επιτευχθεί ένα μεγάλο άλμα σε αυτόν τον συντονισμό.

Οι οντολογίες έχουν δημιουργηθεί για τον διαμοιρασμό και την επαναχρησιμοποίηση της γνώσης και για την απόδοση κάποιου μηχανισμού συλλογισμού των εργασιών που γίνονται σε κάποιο πεδίο ενδιαφέροντος.

Στόχος της παρούσας μεταπτυχιακής διατριβής είναι η μελέτη της τεχνολογίας των οντολογιών, με πεδίο εφαρμογής τους την ιατρική, και πιο συγκεκριμένα την ανάπτυξη μιας οντολογίας στο πεδίο των καρδιαγγειακών ασθενειών (coronary artery disease). Μελετάται η θεωρητική πλευρά των οντολογιών, δηλαδή το τι είναι μια οντολογία, οι διάφοροι ορισμοί που έχουν δοθεί κατά καιρούς, τα συστατικά που απαρτίζουν μια οντολογία, οι διάφορες μεθοδολογίες σχεδιασμού οντολογιών που έχουνε δοθεί, κάποια εργαλεία που χρησιμοποιούνται για την ανάπτυξή τους, καθώς και γλώσσες αναπαράστασής τους, τα είδη και η κατηγοριοποίηση των οντολογιών. Αμέσως μετά γίνεται μια έρευνα για γενικές και διεθνώς αποδεκτές οντολογίες που έχουν σαν πεδίο ενδιαφέροντος την ιατρική και χρησιμοποιούνται για την κατασκευή άλλων οντολογιών, πιο εξειδικευμένων σε κάποιο συγκεκριμένο σημείο της ιατρικής, που στη δική μας περίπτωση, όπως προαναφέρθηκε, είναι το πεδίο των καρδιαγγειακών ασθενειών.

Το επόμενο κομμάτι είναι η σχεδίαση και υλοποίηση της οντολογίας μας. Η οντολογία σχεδιάστηκε από την αρχή με τη βοήθεια ενός γιατρού, έχοντας, όμως, σαν βάση τη γενική οντολογία UMLS (Unified Medical Language System) που είναι ευρέως αποδεκτή. Το UMLS είναι μια οντολογία η οποία δημιουργήθηκε από την ενοποίηση πολλών ιατρικών λεξικών. Η ενοποίηση αυτή προκύπτει με τη δημιουργία ενός σημασιολογικού δικτύου και σχέσεων μεταξύ των κόμβων του δικτύου αυτού. Κάθε όρος του κάθε λεξικού που συμπεριλαμβάνεται στο UMLS, αντιστοιχίζεται σε έναν τουλάχιστον κόμβο του σημασιολογικού αυτού δικτύου και έτσι μέσω αυτού, προκύπτουν και σημασιολογικές σχέσεις μεταξύ των όρων των διαφορετικών ιατρικών λεξικών. Έγινε προσπάθεια να γίνει η αναπαράσταση του πεδίου των καρδιοαγγειακών νοσημάτων, σύμφωνα πάντα με την κρίση του γιατρού, αλλά χρησιμοποιώντας όρους που εμπεριέχονται στο UMLS, ώστε να συμπεριληφθούν όσο το δυνατόν όροι που είναι ευρέως αποδεκτοί και χρησιμοποιούνται από την πλειοψηφία της παγκόσμιας ιατρικής κοινότητας. Επίσης, μέσω του UMLS μπορέσαμε και πήραμε μαζί με τους όρους και τους συνώνυμους όρους που αυτό συμπεριλαμβάνει.

Τέλος, αφού ολοκληρώθηκε η υλοποίηση της οντολογίας πεδίου που προαναφέραμε, σχεδιάστηκε και υλοποιήθηκε ένα σύστημα, που κάνει χρήση της με σκοπό την ανάκτηση εγγράφων. Η προσπάθεια αυτή είχε ως στόχο την αναζήτηση του τρόπου με τον οποίο γίνεται χρήση των οντολογιών αμέσως μετά την υλοποίηση τους και την ενσωμάτωσή τους σε πραγματικά συστήματα που χρησιμοποιούνται στην καθημερινότητα. Το σύστημα που υλοποιήθηκε, είναι ένα σύστημα που μπορεί κάποιος γιατρός να κάνει εισαγωγή κάποιον εγγράφων με λέξεις κλειδιά που τα χαρακτηρίζουν, τοποθετεί τα κείμενα στους κατάλληλους κόμβους της οντολογίας μας, εάν αυτό είναι εφικτό (indexing), δημιουργώντας κάποια σχέση μεταξύ των κειμένων και των αντίστοιχων όρων της οντολογίας με τους οποίους συσχετίζονται, καθώς επίσης επιτρέπει στο γιατρό να κάνει ερωτήσεις στο σύστημα και αυτό με τη σειρά του να του επιστρέψει τα κείμενα που σχετίζονται με αυτές και που έχουν εισαχθεί σε αυτό με την προηγούμενη διαδικασία.

1.2. Δομή της Διατριβής

Η παρούσα διατριβή αποτελείται από 8 κεφάλαια. Στο κεφάλαιο 2 γίνεται μια θεωρητική ανασκόπηση των οντολογιών. Δίνεται μια περιγραφή των οντολογιών καθώς και οι ορισμοί που έχουν δοθεί κατά καιρούς για αυτές. Αναλύονται τα βασικά χαρακτηριστικά τους, αρχές που πρέπει να διέπουν τη σχεδίαση των οντολογιών καθώς και οι διαδικασίες που εμπλέκονται στην ανάπτυξή τους. Δίνονται κάποιες μέθοδοι που υπάρχουν στη βιβλιογραφία, σχετικές με την ανάπτυξη οντολογιών, τα είδη στα οποία μπορούμε να κατατάξουμε μια οντολογία, καθώς και κάποια εργαλεία και γλώσσες που προσανατολίζονται στην ανάπτυξής τους.

Στο κεφάλαιο 3 περιγράφονται κάποιες οντολογίες που έχουν αναπτυχθεί και συμπεριλαμβάνουν γνώση στο πεδίο της ιατρικής. Αυτές είναι η OpenCyc και WordNet που αποτελούν γενικές οντολογίες, αλλά συμπεριλαμβάνουν και γνώση ιατρικής και οι GALEN, UMLS, SNOMED CT και FMA που αποτελούν οντολογίες αφοσιωμένες στο πεδίο της ιατρικής.

Στο κεφάλαιο 4 περιγράφονται διάφορες εργασίες που κάνουν χρήση οντολογιών στο πεδίο της ιατρικής. Οι εργασίες αυτές ομαδοποιούνται ανάλογα με το πρόβλημα που προσπαθούν να επιλύσουν οι συγγραφείς τους. Έτσι έχουμε εργασίες που αφιερώνονται στη χρήση

οντολογιών για σημασιολογική διαλειτουργικότητα συστημάτων, στη χρήση οντολογιών με σκοπό τη σημασιολογική αναζήτηση σε βάσεις δεδομένων, στη χρήση οντολογιών με σκοπό την ανάκτηση πληροφορίας και εγγράφων και στη χρήση και κατασκευή οντολογιών με ταυτόχρονη χρήση Natural Language Processing tools (NLP).

Στο κεφάλαιο 5 παρουσιάζονται τα διάφορα εργαλεία και γλώσσες οντολογιών που χρησιμοποιήθηκαν κατά τη διάρκεια υλοποίησης της μεταπτυχιακής αυτής διατριβής.

Στο κεφάλαιο 6 δίνεται η όλη φάση σχεδίασης και ανάπτυξης της οντολογίας στο πεδίο της καρδιολογίας και πιο συγκεκριμένα στο πεδίο των καρδιαγγειακών νοσημάτων.

Στο κεφάλαιο 7 παρουσιάζεται η σχεδίαση και ανάπτυξη ενός συστήματος βασιζόμενου σε οντολογία, που έχει σαν σκοπό την ανάκτηση εγγράφων με σημασιολογικά κριτήρια.

Στο κεφάλαιο 8 παρουσιάζονται συμπεράσματα που προέκυψαν από τη μεταπτυχιακή διατριβή, καθώς και μελλοντική εργασία που θα μπορούσε να γίνει για τη βελτίωση τόσο της οντολογίας που αναπτύχθηκε αλλά και του συστήματος ανάκτησης εγγράφων.

ΚΕΦΑΛΑΙΟ 2. ΘΕΩΡΗΤΙΚΗ ΑΝΑΣΚΟΠΗΣΗ ΟΝΤΟΛΟΓΙΩΝ

2.1. Εισαγωγή στις Οντολογίες

2.1.1. Τι είναι μια οντολογία;

Για το τι είναι μια «οντολογία», υπάρχουν δύο οπτικές γωνίες, από όπου μπορούμε να αντλήσουμε την απάντηση. Η μία προέρχεται από την επιστήμη της φιλοσοφίας και η άλλη από την επιστήμη της πληροφορικής.

Οπτική γωνία φιλοσοφίας:

Στο λεξικό «Webster» υπάρχει ο εξής ορισμός της οντολογίας:

Παρακλάδι της μεταφυσικής που σχετίζεται με τη φύση και της σχέσεις των όντων.
Μια ιδιαίτερη θεωρία για τη φύση των όντων και τα είδη της ύπαρξης.

Οι αρχαίοι Έλληνες είχαν ασχοληθεί με το ερώτημα: «Ποια είναι η ουσία των πραγμάτων μέσω των διάφορων αλλαγών;». Απαντήσεις έχουν δοθεί σε αυτό το ερώτημα από διαφόρους φιλόσοφους, από τον Παρμενίδη τον Ελεάτη ($5^{ος}$ και $4^{ος}$ αιώνας π.χ.), τον πρόδρομο της οντολογίας, μέχρι τον Αριστοτέλη, συγγραφέα του «Μετά τα φυσικά» (μία εργασία που μπορεί εύκολα να χαρακτηριστεί Οντολογία).

Έτσι οντολογία είναι μια συστηματική περιγραφή της ύπαρξης. Στη φιλοσοφία είναι μια ρητά τυποποιημένη προδιαγραφή για το πώς αναπαριστούμε αντικείμενα, έννοιες και τις οντότητες που θεωρούνται πως υπάρχουν σε μια περιοχή ενδιαφέροντος και οι σχέσεις που υπάρχουν μεταξύ αυτών. Για όλα τα συστήματα το τι «υπάρχει» είναι αυτά τα οποία μπορούν να αναπαρασταθούν. Όταν η γνώση που αφορά μια περιοχή αναπαρίσταται σε μια δηλωτική γλώσσα, το σύνολο των αντικειμένων που μπορούν να αναπαρασταθούν ονομάζεται «ο κόσμος της πραγματείας» (universe of discourse).

Ο Αριστοτέλης διέκρινε διαφορετικές μορφές ύπαρξης για να εντοπίσει ένα σύστημα από κατηγορίες (ύλη, ποιότητα, ποσότητα, σχέση, ενέργεια, συναίσθημα, χώρος και χρόνος), ώστε να ταξινομήσει οτιδήποτε υπάρχει στον κόσμο. Για παράδειγμα όταν λέμε «ο ηλεκτρονικός υπολογιστής είναι στο τραπέζι», υποδηλώνουμε έναν διαφορετικό τρόπο ύπαρξης από όταν λέμε «ο ηλεκτρονικός υπολογιστής είναι γκρι». Η πρώτη δήλωση εντάσσεται στην κατηγορία «χώρος» ενώ η δεύτερη στην κατηγορία «ποιότητα». Η κατηγοριοποίηση που δόθηκε από τον Αριστοτέλη ήταν ευρέως αποδεκτή μέχρι τον $18^{ο}$ αιώνα.

Μία λίγο διαφορετική αντίληψη δόθηκε από τον Emmanuel Kant (1724-1804). Ο Kant πρόσθεσε πως η αίσθηση των πραγμάτων δεν καθορίζεται μόνο από τα ίδια τα πράγματα και μόνο, αλλά και από τη συμβολή όποιου τα αντιλαμβάνεται και τα κατανοεί. Έτσι ο Kant έβαλε ένα νέο ερώτημα: «ποιες δομές χρησιμοποιεί το μυαλό της για να αντιληφθεί την πραγματικότητα;».

Μια ταξινόμηση των κατηγοριών που προαναφέρθηκε, είναι γνωστή στους φιλοσόφους ως μια οντολογία. Από όσα έχουμε προαναφέρει πρέπει να αποσαφηνιστεί πως « μια οντολογία (an ontology)» είναι διαφορετικό από την «οντολογία (ontology)». Το πρώτο είναι μια ταξινόμηση κατηγοριών, ενώ το δεύτερο είναι ένα παρακλάδι της φιλοσοφίας.

Οπτική γωνία επιστήμης της πληροφορικής:

Στην επιστήμη της πληροφορικής, οντολογία είναι μια απόπειρα για τη διατύπωση ενός εξαντλητικού και αυστηρού εννοιολογικού σχήματος, ενός πεδίου ενδιαφέροντος, μια τυπική ιεραρχική δομή που περιέχει όλες τις σχετικές οντότητες και τις σχέσεις μεταξύ τους, καθώς και τους κανόνες που ανήκουν στο πεδίο αυτό.

Μπορούμε να υποθέσουμε πως υπάρχει ένας παραλληλισμός μεταξύ της πραγματικότητας έτσι όπως την αντιλαμβάνονται οι άνθρωποι και οι υπολογιστές, και μαζί μπορούν να ενσωματωθούν σε μια δομή, τις οντολογίες [2]. Σύμφωνα με την ιδέα αυτή, αν ένα κομπιούτερ είναι αποκλειστικά αφοσιωμένο να απαντά σε ερωτήσεις που έχουν να κάνουν με «ταξίδια», η πραγματικότητά του μπορεί να δομηθεί ταξινομώντας τα ταξίδια, σαν ταξίδι με αεροπλάνο, ταξίδι με τρένο κ.α. Ωστόσο για να είναι αυτή η ταξινόμηση μια πραγματική οντολογία για το κομπιούτερ, αυτό πρέπει να είναι ικανό να αντεπεξέρχεται με αυτή, δηλαδή να μπορεί να την κατανοεί. Αυτή είναι η πρώτη σημαντική διαφορά μεταξύ οντολογιών από την φιλοσοφική πλευρά και από την πλευρά της επιστήμης της πληροφορικής. Σύμφωνα με τη δεύτερη οπτική γωνία, μια οντολογία πρέπει να κωδικοποιηθεί σε μια γλώσσα κατανοητή από το μηχάνημα. Με άλλα λόγια όταν ένας ειδικός στις οντολογίες, ορίζει τι είναι μια οντολογία, πρέπει να αλλάζει την προοπτική του αντίληψη από την αντίληψη ενός ανθρώπου στην αντίληψη ενός υπολογιστή. Έτσι αν ένας υπολογιστής δεν μπορεί να κατανοήσει την οντολογία, τότε αυτή δεν μπορεί να είναι και οντολογία. Επιπλέον από την οπτική γωνία της επιστήμης της πληροφορικής, μια οντολογία είναι συνήθως πιο συγκεκριμένη από ότι μια οντολογία με φιλοσοφική προσέγγιση. Τέλος, λόγω της χρήσης του όρου «οντολογία», τα χαρακτηριστικά του διαμοιρασμού και της επαναχρησιμοποίησης έχουν γίνει βασικά στον ορισμό αυτού του όρου από την πλευρά ενός μηχανικού. Τέτοια χαρακτηριστικά δεν είναι βασικά στις φιλοσοφικές οντολογίες.

2.1.2. Ορισμοί

Ο ορισμός που έχει δώσει ο Thomas R. Gruber για την οντολογία είναι ο εξής:
Μια ρητή προδιαγραφή ενός πεδίου αντίληψης (conceptualization) με το επίκεντρό του στη γνώση και τον διαμοιρασμό [2].

Ο Guarino όρισε την οντολογία ως:
Ένα σύνολο λογικών αξιωμάτων, που δίνονται για να αποδώσουν το επιδιωκόμενο νόημα ενός λεξικού [3].

Ο ορισμός του Sowa:
Το αντικείμενο της οντολογίας είναι η μελέτη των κατηγοριών στις οποίες ανήκουν τα αντικείμενα που υπάρχουν ή μπορεί να υπάρχουν σε κάποιο πεδίο ενδιαφέροντος. Το αποτέλεσμα μιας τέτοιας μελέτης, που ονομάζεται οντολογία, είναι ένας κατάλογος από τύπους αντικειμένων που θεωρείται πως υπάρχουν σε ένα πεδίο ενδιαφέροντος D με την μελλοντική δυνατότητα ενός ατόμου που χρησιμοποιεί μία γλώσσα L για το σκοπό να μιλήσει για το D [4].

Σύμφωνα με τον Barry Smith:
Μια οντολογία είναι κατά μια προσέγγιση, ένας πίνακας κατηγοριών, και κάθε τύπος οντοτήτων αποδίδεται από έναν κόμβο σε μια ιεραρχική δομή [5].

Έτσι μπορούμε να δοθεί ο εξής γενικός ορισμός για έναν μηχανικό οντολογιών [86]**:**
Μία οντολογία, είναι μια τυποποιημένη, ρητή προδιαγραφή, ενός διαμοιραζόμενου πεδίου αντίληψης. Το πεδίο αντίληψης αναφέρεται σε ένα αφηρημένο μοντέλο ενός φαινομένου που υπάρχει στον κόσμο, και έχουν εντοπιστεί οι σχετικές έννοιες αυτού του φαινομένου. Ρητή σημαίνει, πως οι έννοιες που χρησιμοποιούνται, και οι περιορισμοί στη χρήση τους, έχουν ρητά οριστεί. Ο όρος τυποποιημένη, αναφέρεται στο γεγονός πως η οντολογία πρέπει να είναι κατανοητή από υπολογιστή. Τέλος, ο όρος διαμοιραζόμενος αποδίδει την έννοια πως μια οντολογία συλλαμβάνει ομόφωνη γνώση, που δε δίνεται από ένα μεμονωμένο άτομο, αλλά είναι αποδεκτή από μια ομάδα.

Ένας εναλλακτικός ορισμός που δόθηκε από τους Neches and colleagues [6] είναι:

Μία οντολογία ορίζει τους βασικούς όρους και σχέσεις που αποδίδουν το λεξικό μιας τοπικής περιοχής ενδιαφέροντος, όπως επίσης και τους κανόνες για τον συνδυασμό αυτών των όρων και σχέσεων με σκοπό τον ορισμό επεκτάσεων του λεξικού.

2.2. Τα Βασικά Συστατικά μιας Οντολογίας

Υπάρχουν διάφορες αναπαραστάσεις τυποποίησης (και αντίστοιχες γλώσσες) για την τυποποίηση (και υλοποίηση) των οντολογιών. Κάθε μια από αυτές παρέχει διαφορετικά συστατικά που μπορούν να χρησιμοποιηθούν για αυτό το σκοπό. Παρόλα αυτά μοιράζονται το επόμενο ελάχιστο σύνολο συστατικών.

Κλάσεις (Classes) που αναπαριστούν τις έννοιες (*concepts*). Τα *concepts* είναι έννοιες με ευρεία σημασία. Για παράδειγμα στην περιοχή των «ταξιδιών», concepts μπορεί να είναι: τοποθεσίες (πόλεις, χωριά κ.α.), καταλύματα (ξενοδοχεία, κατασκηνώσεις κ.α.) και μέσα μεταφοράς (αεροπλάνα, τρένα, αυτοκίνητα, μοτοσικλέτες και πλοία). Οι κλάσεις στην οντολογία οργανώνονται συνήθως σε ταξινομίες όπου μπορούν να εφαρμοστούν μηχανισμοί κληρονομικότητας. Μπορούμε να αναπαραστήσουμε μια ταξινομία από χώρους ψυχαγωγίας (θέατρο, κινηματογράφος, κονσέρτο κ.α.) ή ταξιδιωτικά πακέτα (οικονομικό, επαγγελματικό κ.α.).

Σχέσεις (relations) που αναπαριστούν έναν τύπο συσχέτισης μεταξύ των concepts. Οι οντολογίες συνήθως περιέχουν δυαδικές σχέσεις. Το πρώτο όρισμα είναι το πεδίο της σχέσης (domain of relation) και το δεύτερο η εμβέλεια της (range). Έτσι η σχέση «τόπος άφιξης (arrivalPlace)» έχει το concept «ταξίδι (Travel)» σαν domain και το concept «τοποθεσία (Place)» σαν εμβέλεια. Για παράδειγμα για να εκφράσουμε ότι η πτήση AA7462-March-10-1007 πηγαίνει στο Seattle, πρέπει να γράψουμε: (arrivalPlace AA7462-March-10-1007 Seattle).

Δυαδικές σχέσεις μερικές φορές χρησιμοποιούνται για να εκφράσουν τα χαρακτηριστικά (attributes) των concepts. Τα ***Attributes*** συνήθως διακρίνονται από τις relations επειδή η εμβέλεια (range) τους είναι κάποιος τύπος δεδομένων (datatype), όπως αλφαριθμητικό, αριθμός, κ.α., ενώ η εμβέλεια (range) μιας σχέσης είναι concept. Έτσι π.χ. το flightNumber

είναι string. Επίσης μπορούμε να εκφράσουμε relations μεγαλύτερου βαθμού όπως «ένας δρόμος μπορεί να συνδέει δύο διαφορετικές πόλεις».

Σύμφωνα με τον Gruber [2], τα ***formal axioms*** βοηθούν στη μοντελοποίηση προτάσεων που είναι πάντα αληθείς. Συνήθως χρησιμοποιούνται για να αναπαραστήσουν γνώση που δεν μπορεί να ορισθεί από τα υπόλοιπα συστατικά. Επιπρόσθετα τα formal axioms χρησιμοποιούνται για να επιβεβαιωθεί η συνέπεια της οντολογίας ή η συνέπεια της γνώσης που είναι αποθηκευμένη σε μια βάση γνώσης (knowledge base). Τα formal axioms είναι πολύ χρήσιμα στον συμπερασμό νέας γνώσης. Ένα formal axiom στην περιοχή των ταξιδιών θα ήταν, ότι είναι αδύνατον να ταξιδέψεις από τη Βόρειο Αμερική στην Ευρώπη με τρένο.

Τα στιγμιότυπα (***instances***) χρησιμοποιούνται για να αναπαραστήσουν οντότητες (elements) ή άτομα (individuals) στην οντολογία. Για παράδειγμα ένα instance για το concept AA7462 είναι η πτήση AA7462 που φθάνει στο Seattle τον Μάρτιο 10, 2007 και κοστίζει 300 (euros, US dollars ή οποιοδήποτε άλλο νόμισμα).

2.3. Αρχές στη Σχεδίαση Οντολογιών

Υπάρχουν κάποια κριτήρια σχεδίασης και ένα σύνολο αρχών που έχουν αποδειχθεί χρήσιμα στη διαδικασία ανάπτυξης οντολογιών. Σύμφωνα με το [7] οι αρχές σχεδίασης μιας οντολογίας είναι κάποια αντικειμενικά κριτήρια για καθοδήγηση και αξιολόγηση της σχεδίασης μιας οντολογίας. Έτσι ορίσθηκαν τα εξής πέντε κριτήρια:

Σαφήνεια (Clarity) [7] , που ορίζεται ως εξής: Μια οντολογία πρέπει να εκφράζει αποτελεσματικά το επιδιωκόμενο νόημα των όρων που ορίζει. Οι ορισμοί πρέπει να δηλώνονται με formal axioms, και ένας ολοκληρωμένος ορισμός (που ορίζεται από αναγκαίες και επαρκείς συνθήκες) προτιμάται από έναν μερικό ορισμό (που ορίζεται μόνο με αναγκαίες ή επαρκείς συνθήκες). Όλοι οι ορισμοί πρέπει να κρατούνται σε έγγραφα σε φυσική γλώσσα.

Ελάχιστη εξειδίκευση κωδικοποίησης (minimal encoding bias) [7], που σημαίνει πως το μοντέλο που πάμε να δημιουργήσουμε, με το οποίο μοντελοποιούμε ένα φαινόμενο, πρέπει να εξαρτάται μόνο από το επίπεδο της γνώσης και όχι από το επίπεδο συμβόλων κάποιας κωδικοποίησης. Αυτό γίνεται για χάρη του διαμοιρασμού της γνώσης, αφού οι διάφοροι

εμπλεκόμενοι στο διαμοιρασμό της γνώσης μπορεί να είναι υλοποιημένοι με διαφορετικούς τρόπους.

Επεκτασιμότητα (extendibility) [7], που μας λέει πως κάποιος πρέπει να μπορεί να είναι σε θέση να ορίσει νέους όρους για ειδικές χρήσεις, βασιζόμενος στο υπάρχον λεξικό, με έναν τρόπο που να μη χρειάζεται η αναθεώρηση των υπαρχόντων ορισμών.

Συνέπεια (coherence) [7], που ορίζει πως μια οντολογία πρέπει να επιτρέπει συμπερασμούς που να είναι συνεπής με τους ορισμούς. Αν συμπεραθεί μια πρόταση που αντικρούει κάποιον ορισμό ή κάποιο παράδειγμα που έχει δοθεί άτυπα, τότε η οντολογία είναι ασυνεπής.

Ελάχιστη οντολογική αφοσίωση (minimal ontological commitment) [7], αφού η αφοσίωση στην οντολογία βασίζεται στη συνεπή χρήση του λεξικού, η οντολογική αφοσίωση μπορεί να ελαχιστοποιηθεί ορίζοντας την πιο αδύναμη θεωρία και μόνο τους όρους που είναι βασικοί για την επικοινωνία της γνώσης με συνέπεια στη θεωρία. Έτσι π.χ. δεν πρέπει να συμφωνήσουμε σε κάποιον συγκεκριμένο τύπο νομίσματος όταν σχεδιάζουμε μια οντολογία, αφού μια τέτοια λεπτομέρεια μπορεί να διαφέρει σε διαφορετικά συστήματα.

Υπάρχουν και άλλα κριτήρια που είναι χρήσιμα στη σχεδίαση οντολογιών, όπως η ***τυποποίηση των ονομάτων (standardization of names)*** [8], που προτείνει τη χρήση κάποιον συμβάσεων για όρους που σχετίζονται μεταξύ τους, για την ευκολότερη κατανόηση της οντολογίας.

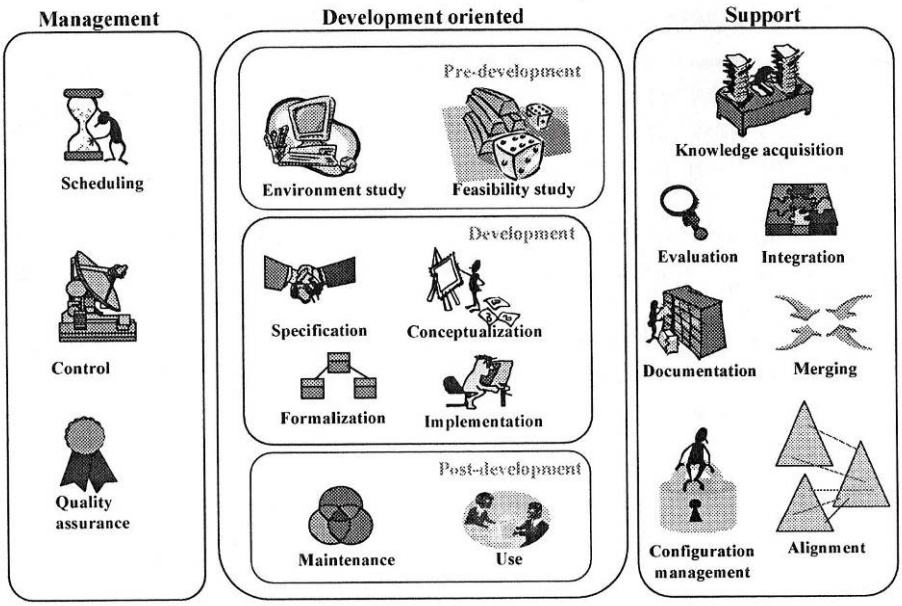
2.4. Διαδικασία Ανάπτυξης μιας Οντολογίας και ο Κύκλος Ζωής της (Life Cycle)

Το 1997 στο πλαίσιο εργασίας του METHODOLOGY [9], που αποτελεί μια μεθοδολογία για την κατασκευή οντολογιών, προσδιορίσθηκε η διαδικασία ανάπτυξης μιας οντολογίας. Η πρόταση αυτή βασίστηκε στο πρότυπο της IEEE για ανάπτυξη λογισμικού [10]. Η διαδικασία ανάπτυξης οντολογιών αναφέρεται στις δραστηριότητες που πρέπει να εκτελεστούν όταν χτίζονται οντολογίες. Αυτές μπορούν να ταξινομηθούν σε τρεις κατηγορίες όπως φαίνεται στο Σχήμα 2.1.

Δραστηριότητες διαχείρισης οντολογίας (Ontology management activities), που περιλαμβάνουν τον χρονοπρογραμματισμό (**scheduling**), τον έλεγχο (**control**) και την ασφάλεια ποιότητας (**quality assurance**). Η δραστηριότητα χρονοπρογραμματισμού αναγνωρίζει τις εργασίες που πρέπει να πραγματοποιηθούν, τη διάταξη η οποία πρέπει να ακολουθηθεί, και τον χρόνο και τους πόρους που χρειάζονται για την ολοκλήρωσή τους. Η δραστηριότητα αυτή είναι πολύ βασική για οντολογίες που χρησιμοποιούν άλλες αποθηκευμένες σε βιβλιοθήκες οντολογιών ή για οντολογίες που πρέπει να είναι πολύ γενικές. Η διαδικασία ελέγχου εγγυάται πως οι χρονοπρογραμματισμένες εργασίες έχουν ολοκληρωθεί, έτσι όπως είχε σχεδιαστεί να γίνει. Τέλος η ασφάλεια ποιότητας, σιγουρεύει πως η ποιότητα κάθε προϊόντος που εξάχθηκε (οντολογία, λογισμικό και τα έγγραφα τεκμηρίωσης) είναι ικανοποιητικά.

Δραστηριότητες στραμμένες στην ανάπτυξη οντολογιών (Ontology development-oriented activities) που ομαδοποιούνται στην προ-ανάπτυξης δραστηριότητα (pre-development), δραστηριότητα ανάπτυξης (development) και μετά-ανάπτυξης δραστηριότητα (post-development).

Κατά τη διάρκεια της προ-ανάπτυξης, γίνεται μια μελέτη του περιβάλλοντος (**environment study**) και εντοπίζεται το πρόβλημα που θα λυθεί με την οντολογία, οι εφαρμογές στις οποίες θα ενσωματωθεί η οντολογία κ.α. Επίσης κατά τη διάρκεια αυτής της δραστηριότητας, η μελέτη εφικτότητας (**feasibility study**) απαντά σε ερωτήματα όπως «είναι δυνατόν να υλοποιηθεί η οντολογία;», «είναι κατάλληλο να υλοποιηθεί η οντολογία;» κ.α.



Σχήμα 2.1 Διαδικασία Ανάπτυξης μιας Οντολογίας. [9]

Στη φάση της ανάπτυξης, έχουμε τέσσερις επιπλέον δραστηριότητες. Στη δραστηριότητα των προδιαγραφών (**specification activity**), ορίζεται ο σκοπός για τον οποίο χτίζεται η οντολογία, ποιος είναι ο επιδιωκόμενος στόχος της και ποιοι θα είναι οι τελικοί χρήστες. Στη δραστηριότητα προσδιορισμού γνώσεων και εμπειριών (**conceptualization activity**), δομείται η γνώση του πεδίου του ενδιαφέροντος, σαν ένα εννοιολογικό μοντέλο, είτε από την αρχή είτε χρησιμοποιώντας ήδη υπάρχοντα μοντέλα. Στη δεύτερη περίπτωση, άλλες σχετικές δραστηριότητες, πριονίζουν τα κλαδιά των υπαρχόντων οντολογιών, επεκτείνουν την οντολογία στα υψηλά επίπεδα για να επιτύχουν την κάλυψη πιο γενικών εννοιών, ή εξειδικεύουν κλαδιά τους που χρειάζονται να αναλυθούν περισσότερο. Δοθέντος ότι η δραστηριότητα του conceptualization είναι ανεξάρτητη από τη γλώσσα υλοποίησης τους, επιτρέπει τη μοντελοποίηση των οντολογιών σύμφωνα με το κριτήριο της ελάχιστης εξειδίκευσης κωδικοποίησης (minimal encoding bias). Η δραστηριότητα τυποποίησης (**formalization activity**), μετατρέπει το εννοιολογικό μοντέλο σε ένα πιο τυποποιημένο (semi-computable). Τέλος, σε αυτή τη φάση έχουμε τη δραστηριότητα υλοποίησης (**implementation activity**) όπου υλοποιείται ένα μοντέλο πλήρως αναγνώσιμο από υπολογιστή (computable model) σε μια γλώσσα υλοποίησης οντολογιών.

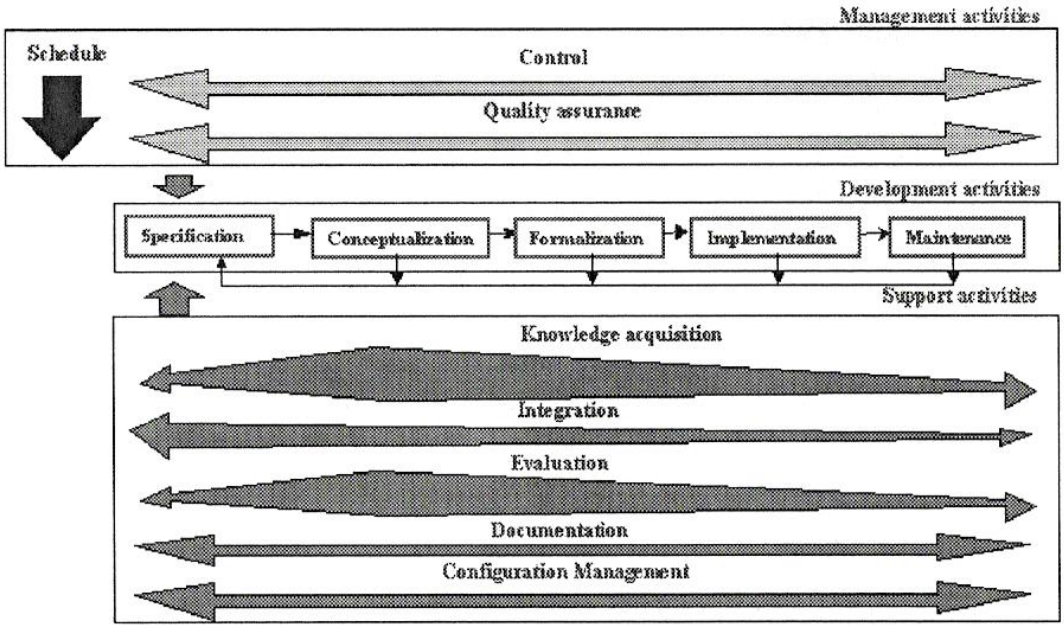
Κατά τη φάση της μετά-ανάπτυξης, η δραστηριότητα της συντήρησης (**maintenance activity**) κρατάει ενήμερη την οντολογία και τη διορθώνει αν χρειάζεται. Επίσης σε αυτή τη φάση η οντολογία είναι έτοιμη για χρησιμοποίηση από εφαρμογές ή επαναχρησιμοποίηση από άλλες οντολογίες. Τέλος εδώ πρέπει να αναφερθεί και η ύπαρξη της δραστηριότητας εξέλιξης της οντολογίας (**evolution activity**), που αποτελείται από την διαχείριση των αλλαγών που υφίσταται μια οντολογία, και από τη δημιουργία και συντήρηση διάφορων παραλλαγών της οντολογίας, γιατί αυτές μπορεί είτε να χρησιμοποιηθούν από άλλες εφαρμογές είτε να επαναχρησιμοποιηθούν στη δημιουργία άλλων οντολογιών.

Τέλος οι **δραστηριότητες υποστήριξης μιας οντολογίας (ontology support activities)**, περιέχουν μια σειρά δραστηριοτήτων που μπορούν να εκτελεστούν κατά τη διάρκεια των δραστηριοτήτων που είναι στραμμένες προς την ανάπτυξη οντολογιών (Ontology development-oriented activities), και χωρίς αυτές μια οντολογία δε θα μπορούσε να πραγματοποιηθεί. Περιλαμβάνονται οι εξής δραστηριότητες: απόκτηση γνώσης (**knowledge acquisition**), αξιολόγηση (**evaluation**), ενσωμάτωση (**integration**), συγχώνευση (**merging**), ευθυγράμμιση (**alignment**), δημιουργία εγγράφων σχετικών με την όλη διαδικασία δημιουργίας της οντολογίας (**documentation**) και τη διαχείριση διάρθρωσης (**configuration management**). Στόχος της δραστηριότητας απόκτησης γνώσης (knowledge acquisition) είναι να αποκτηθεί μέσω ειδικών, γνώση για το πεδίο του ενδιαφέροντος ή μέσω κάποιας ημιαυτόματης διαδικασίας, που ονομάζεται **ontology learning** [11]. Η δραστηριότητα αξιολόγησης (evaluation) [12], κάνει μια τεχνική κρίση των οντολογιών, των λογισμικών που σχετίζονται με αυτές καθώς και των κειμένων (documentation) που τα συνοδεύει. Η κρίση αυτή γίνεται σε κάποια σημεία κατά τη διάρκεια των διάφορων σταδίων, καθώς και ανάμεσά στα διάφορα στάδια του κύκλου ζωής της οντολογίας. Η δραστηριότητα ενσωμάτωσης (integration) είναι απαραίτητη όταν δημιουργούμε μια νέα οντολογία χρησιμοποιώντας οντολογίες που είναι ήδη διαθέσιμες. Επίσης άλλη μια δραστηριότητα υποστήριξης είναι η συγχώνευση (merging), η οποία αποτελείται από τη δημιουργία μιας νέας οντολογίας ξεκινώντας από άλλες οντολογίες του ίδιου πεδίου ενδιαφέροντος. Η οντολογία που προκύπτει είναι ικανή να ενοποιεί έννοιες (concepts), ορολογίες (terminologies), ορισμούς (definitions), περιορισμούς (constraints) κ.α. από όλες τις αρχικές οντολογίες. Η δραστηριότητα ευθυγράμμισης (alignment) εντοπίζει διαφορετικά είδη αντιστοίχησης (mappings) ή συνδέσεις (links) μεταξύ των εμπλεκόμενων οντολογιών. Έτσι, η δραστηριότητα αυτή διατηρεί τις αρχικές οντολογίες και δεν τις συγχωνεύει. Στη

δραστηριότητα δημιουργίας εγγράφων σχετικών με την όλη διαδικασία δημιουργίας της οντολογίας (documentation), εκθέτονται λεπτομερώς και με σαφήνεια όλα τα στάδια που έχουν ολοκληρωθεί καθώς και τα προϊόντα που έχουν παραχθεί. Με τη διαχείριση διάρθρωσης (configuration management) καταγράφονται όλες οι εκδώσεις της προηγούμενης δραστηριότητας (documentation) και του κώδικα της οντολογίας για να ελεγχθούν οι αλλαγές που έχουν γίνει. Η δραστηριότητα της πολυγλωσσίας (multilingualism activity) αναλαμβάνει την αντιστοίχηση της οντολογίας σε διάφορες γλωσσολογικές περιγραφές [13]. Συνήθως η δραστηριότητα αυτή δεν περιλαμβάνεται στις δραστηριότητες υποστήριξης, αλλά είναι πιο σχετική με δικτυακές οντολογίες που είναι διαθέσιμες στο εννοιολογικό δίκτυο (semantic web).

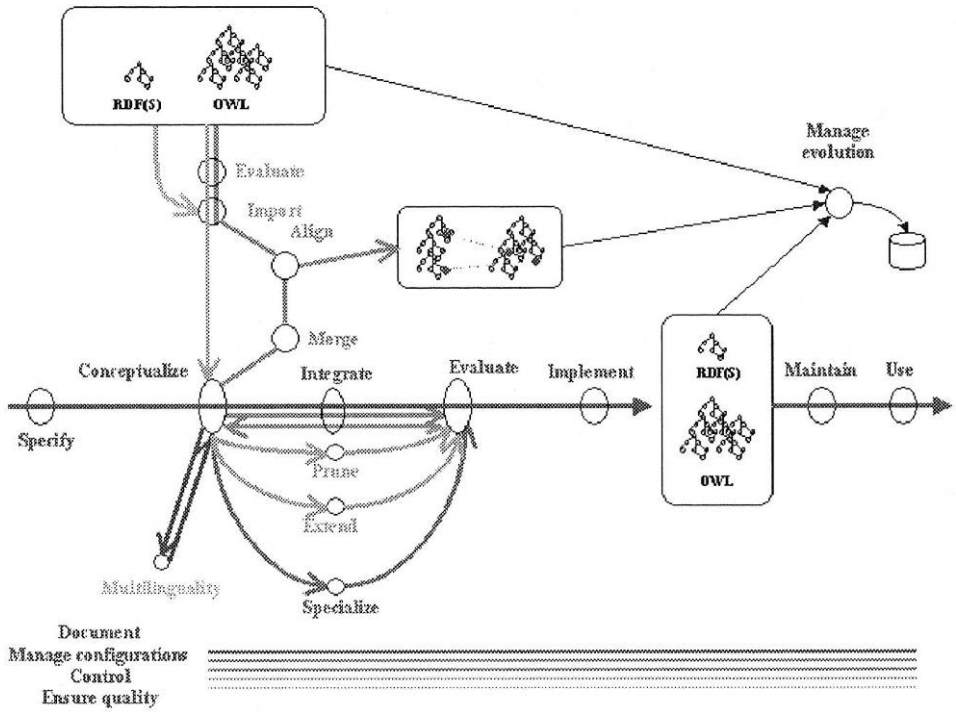
Όπως βλέπουμε η διαδικασία ανάπτυξης μιας οντολογίας δε συμπεριλαμβάνει τη διάταξη με την οποία οι διάφορες δραστηριότητες πρέπει να πραγματοποιηθούν. Αυτός είναι ο ρόλος του κύκλου ζωής της οντολογίας (**ontology life cycle**) [10], ο οποίος παραθέτει πότε οι δραστηριότητες πρέπει να πραγματοποιηθούν. Έτσι, μας δίνει το σύνολο των σταδίων μέσω των οποίων κινείται η οντολογία κατά τη διάρκεια του χρόνου ζωής της, περιγράφοντας ποιες δραστηριότητες πρέπει να πραγματοποιηθούν σε κάθε στάδιο και πως τα στάδια αυτά σχετίζονται μεταξύ τους.

Η αρχική έκδοση του κύκλου ζωής που μοντελοποιεί την METHODOLOGY (Σχήμα 2.2), προτείνει η αρχή να γίνεται με την δραστηριότητα του χρονοπρογραμματισμού (scheduling) των δραστηριοτήτων που πρέπει να πραγματοποιηθούν. Μετά ξεκινά η δραστηριότητα ορισμού των προδιαγραφών (specification activity), δείχνοντας γιατί κατασκευάζεται η οντολογία, ποια θα είναι η πιθανή χρήση της και ποιοι θα είναι οι τελικοί χρήστες της. Μόλις τελειώσει η προηγούμενη δραστηριότητα, ξεκινά ο προσδιορισμός γνώσεων και εμπειριών (conceptualization activity). Σκοπός της είναι να οργανώσει και να δομήσει τη γνώση που αποκτήθηκε στη δραστηριότητα απόκτησης γνώσης (knowledge acquisition), χρησιμοποιώντας ένα σύνολο από αναπαραστάσεις που μπορούν εύκολα οι ειδικοί του πεδίου ενδιαφέροντος να μεταχειριστούν. Αφού το εννοιολογικό μοντέλο έχει δημιουργηθεί, η METHODOLOGY προτείνει την αυτόματη υλοποίηση της οντολογίας χρησιμοποιώντας μεταφραστές (translators). Περισσότερες λεπτομέρειες μπορούν να βρεθούν στο [14].



Σχήμα 2.2 Ο Κύκλος Ζωής μιας Οντολογίας στο METHODOLOGY. [10]

Όσο περισσότερες οντολογίες γίνονται διαθέσιμες σε βιβλιοθήκες οντολογιών ή είναι διασκορπισμένες σε ολόκληρο το ίντερνετ, η επαναχρησιμοποίηση τους από άλλες οντολογίες και εφαρμογές αυξάνει. Οντολογίες διάφορων πεδίων ενδιαφέροντος, μπορούν να επαναχρησιμοποιηθούν για τη δημιουργία άλλων, πιο αναλυτικών και με μεγαλύτερη κάλυψη των πεδίων αυτών, ή μπορεί να συγχωνεύονται με άλλες για τη δημιουργία καινούριων. Στο Σχήμα 2.3 βλέπουμε διάφορους τρόπους για την κατασκευή οντολογιών. Μπορεί να σημειωθεί ότι στο Σχήμα βλέπουμε μια βασική γραμμή (στο μέσω του Σχήματος), άλλες γραμμές που αρχίζουν από αυτή ή τελειώνουν σε αυτή, ή γραμμές που τρέχουν παράλληλα και διακλαδώνονται σε κάποιο σημείο. Έτσι σχέσεις αλληλεξάρτησης (interdependence relationships) [15] υπάρχουν μεταξύ των κύκλων ζωής πολλών οντολογιών, και ενέργειες αξιολόγησης (evaluation), κλαδέματος (pruning) και συγχώνευσης (merging) μπορούν να εφαρμοστούν σε αυτές τις οντολογίες. Οι κύκλοι ζωής οντολογιών διασταυρώνονται και δημιουργούνται διάφορα σενάρια με διαφορετικές τεχνολογικές απαιτήσεις. Παρακάτω θα δώσουμε μερικά από τα πιο συνηθισμένα σενάρια που μπορούν να παρουσιαστούν κατά τη διάρκεια ανάπτυξης οντολογιών.



Σχήμα 2.3 Τρόποι Ανάπτυξης Οντολογιών. [15]

2.5. Σενάρια που Μπορούν να Παρουσιαστούν κατά τη Διαδικασία Ανάπτυξης μιας Οντολογίας

Σενάριο 1ο: **Αξιολόγηση + εισαγωγή (evaluate + import).** Η εισαγωγή μιας οντολογίας αποτελείται από την ενσωμάτωση μια οντολογίας που είναι διαθέσιμη σε μία γλώσσα ή εργαλείο σε κάποιο άλλο εργαλείο οντολογιών. Συνήθως υπάρχουν πολλές υποψήφιες οντολογίες, υλοποιημένες σε διαφορετικές γλώσσες που μπορούν να επαναχρησιμοποιηθούν. Σε αυτήν την περίπτωση είναι βασικό να αξιολογήσουμε το περιεχόμενό τους και την αναλυτικότητά τους, να τις συγκρίνουμε και να επιλέξουμε την καλύτερη για κάθε περίπτωση. Επίσης, είναι σημαντικό να ελέγξουμε την εκφραστικότητα της γλώσσας με την οποία είναι υλοποιημένη η οντολογία, αφού είναι πιθανόν να χαθεί σημαντικό μέρος της γνώσης που περιέχει μια οντολογία αν το μοντέλο γνώσης που ακολουθεί το εργαλείο οντολογιών «στόχος» που προσπαθούμε να την εισάγουμε είναι λιγότερο εκφραστικό από αυτό της γλώσσας ή του εργαλείου με το οποίο είναι υλοποιημένη.

Σενάριο 2ο: Σύλληψη των εννοιών + ενσωμάτωση + αξιολόγηση προσδιορισμού γνώσεων και εμπειριών (conceptualize + integrate + evaluate conceptualization). Αφού μια οντολογία έχει εισαχθεί, το επόμενο βήμα είναι να ενσωματώσουμε το εννοιολογικό μοντέλο της στο εννοιολογικό μοντέλο της οντολογίας που είναι υπό κατασκευή. Αυτό συνεπάγεται πως οι δραστηριότητας της ενσωμάτωσης (integration) και αξιολόγησης (evaluation) του προσδιορισμού γνώσεων και εμπειριών (conceptualization) βρίσκονται στη βασική γραμμή του κύκλου ζωής.

Σενάριο 3ο: **Σύλληψη των εννοιών + απόκτηση γνώσης (conceptualize + acquire knowledge).** Αφού η οντολογία αξιολογήθηκε, εισάχθηκε και ενσωματώθηκε στο εννοιολογικό μοντέλο της κύριας οντολογίας, πρέπει κάποιος να δει τι περιλαμβάνεται στις απαιτήσεις (requirement specification document) της οντολογίας που είναι υπό υλοποίηση και να ελέγξει αν χρειάζεται να γίνει κάτι από τα παρακάτω:

Πριόνισμα (pruning) κλαδιών της οντολογίας που δεν κρίνονται απαραίτητα αφού δεν περιλαμβάνονται στο έγγραφο προσδιορισμού απαιτήσεων.

Εξειδίκευση των κλαδιών που θεωρούνται πως χρειάζονται περισσότερη αναλυτικότατα, περιλαμβάνοντας πιο εξειδικευμένες έννοιες (concepts) και σχέσεις (relations).

Επέκταση της οντολογίας εισάγοντας νέες έννοιες και σχέσεις.

Αναζήτηση άλλων οντολογιών που συμπληρώνουν τις ελλείψεις που βρέθηκαν.

Αν αυτός που δημιουργεί την οντολογία κάνει κάτι από τα παραπάνω, μπορεί να χρειαστεί κάποια επιπλέον γνώση που μπορεί να αποκτηθεί χρησιμοποιώντας κλασικές μεθόδους και τεχνικές εφαρμοσμένης μηχανικής γνώσης (knowledge engineering), ή ημιαυτόματες μεθόδους μάθησης οντολογιών από κείμενα και άλλους πόρους.

Σενάριο 4ο: **Ημιαυτόματη κατασκευή οντολογιών (semi-automatic construction of ontologies)**. Η μάθηση οντολογιών είναι η διαδικασία που μερικώς αυτοματοποιεί την κατασκευή οντολογιών χρησιμοποιώντας μερικές από τις παρακάτω μεθόδους, τεχνικές και εργαλεία: ανάλυση της φυσικής γλώσσας (natural language analysis), στατιστικές μεθόδους

(statistical methods), γλωσσικά πρότυπα (linguistic partners), εξόρυξη κειμένου (text mining) κ.α. Αυτή η διαδικασία χρησιμοποιεί κείμενα, ηλεκτρονικά λεξικά, γλωσσολογικές οντολογίες (όπως η WordNet), και δομημένες πηγές πληροφορίας και γνώσης.

Σενάριο 5ο: **Αξιολόγηση και εισαγωγή ενός συνόλου οντολογιών, και ευθυγράμμισή τους (align)**. Αρκετά συχνά βρίσκονται οντολογίες που μοντελοποιούν το ίδιο πεδίο ενδιαφέροντος. Υπάρχουν περιπτώσεις που θέλουμε να συγκρίνουμε οντολογίες του ίδιου πεδίου και να βρούμε ποιοι όροι της μιας οντολογίας αντιστοιχούν σε όρους μιας άλλης. Οι αντιστοιχήσεις μεταξύ οντολογιών που απορρέουν από αυτή τη διαδικασία ονομάζονται χαρτογραφήσεις (mappings). Έτσι εντοπίζονται σχέσεις μεταξύ δύο οντολογιών. Επίσης υπάρχουν περιπτώσεις που βρίσκονται σχέσεις μεταξύ οντολογιών διαφορετικής κατηγορίας, όπως την περίπτωση της ένωσης μιας οντολογίας ενός πεδίου ενδιαφέροντος (domain ontology) με μια οντολογία υψηλότερου επιπέδου (high level ontology). Κάθε ευθυγράμμιση (alignment) χρειάζεται αξιολόγηση.

Σενάριο 6ο: **Αξιολόγηση και εισαγωγή ενός συνόλου οντολογιών και συγχώνευσή τους (evaluate and import a set of ontologies, and merge them)**. Αυτό είναι μια επέκταση του 5ου σεναρίου. Αφού οι χαρτογραφήσεις (mappings) μεταξύ των οντολογιών είναι γνωστές ο μηχανικός οντολογιών μπορεί να τις συγχωνεύσει σε μία καινούρια οντολογία.

Σενάριο 7ο: Μετάφραση της οντολογίας σε μία άλλη φυσική γλώσσα (Ισπανικά, Αγγλικά, Γαλλικά κ.α.).

Σενάριο 8ο: **Διαχείριση της εξέλιξης της οντολογίας (manage the evolution of the ontology)**. Δοθέντος ότι βασικό χαρακτηριστικό των οντολογιών, είναι η ομοφωνία, ο ποιο φυσικός τρόπος για την ανάπτυξή τους είναι μέσω της συνεργασίας. Μηχανικοί οντολογιών δουλεύουν παράλληλα στην ίδια οντολογία και υπάρχει ανάγκη να διατηρούν και να συγκρίνουν διαφορετικές εκδόσεις, να ελέγχουν τις αλλαγές που άλλοι κάνανε, και να αποδεχτούν ή όχι τις αλλαγές.

Σενάριο 9ο: **Εκτέλεση δραστηριοτήτων υποστήριξης (perform support activities)**. Οι δραστηριότητες της δημιουργίας εγγράφων σχετικών με όλη τη διαδικασία δημιουργίας της οντολογίας (documentation), της διαχείρισης της εξέλιξης (evolution management), της

διαχείρισης διάρθρωσης (configuration management) και διασφάλισης ποιότητας (quality assurance) και ελέγχου (control) εκτελούνται σε όλη τη διάρκεια της διαδικασίας ανάπτυξης μιας οντολογίας.

2.6. Μέθοδοι και Μεθοδολογίες

Σε αυτή την ενότητα θα παρατεθούν κάποιες βασικές μέθοδοι και μεθοδολογίες που έχουν δοθεί κατά καιρούς για την υλοποίηση οντολογιών είτε από την αρχή είτε χρησιμοποιώντας άλλες οντολογίες. Αυτές είναι η Cyc μέθοδος, η Uschold και King μέθοδος, η μεθοδολογία Gruninger και Fox, η προσέγγιση KACTUS, η METHODOLOGY, η SENSUS μέθοδος και η On-To-Knowledge μεθοδολογία.

Η μέθοδος που χρησιμοποιήθηκε για την υλοποίηση της **Cyc** βάσης γνώσης [16] αποτελείται από τρεις φάσεις. Η πρώτη φάση αποτελείται από την εξαγωγή με το χέρι κοινής γνώσης που είναι αυτονόητη σε διάφορους πόρους. Η δεύτερη και τρίτη φάση αποτελούνται από την απόκτηση κοινής γνώσης χρησιμοποιώντας εργαλεία φυσικής γλώσσα ή μηχανικής μάθησης. Η διαφορά μεταξύ των δύο αυτών φάσεων είναι ότι η δεύτερη φάση μπορεί να υποβοηθάται από εργαλεία, άλλα κυρίως γίνεται από ανθρώπους, ενώ στην τρίτη φάση η απόκτηση της κοινής γνώσης γίνεται κυρίως από εργαλεία. Οι οντολογίες που γίνονται σύμφωνα με αυτή τη μέθοδο υλοποιούνται στην γλώσσα CycL.

Η μέθοδος **Uschold και King** [17] προτείνει τέσσερις φάσεις. (1) Ορισμός του σκοπού της οντολογίας, (2) Υλοποίηση της οντολογίας, (3) Αξιολόγησή της οντολογίας και (4) τεκμηρίωσή της. Κατά τη διάρκεια της υλοποίησης, οι συγγραφείς προτείνουν τη σύλληψη της γνώσης, την κωδικοποίησή της και την ενσωμάτωση άλλων οντολογιών στην τρέχουσα. Επίσης, οι συγγραφείς προτείνουν τρεις στρατηγικές για την εύρεση των βασικών εννοιών (concepts) στην οντολογία: μια από πάνω προς τα κάτω προσέγγιση (top-down approach), στην οποία οι πιο αφηρημένες έννοιες (concepts) βρίσκονται και μετά εξειδικεύονται σε πιο συγκεκριμένες. Μια από κάτω προς τα πάνω προσέγγιση (bottom-up approach) στην οποία οι πιο εξειδικευμένες έννοιες (concepts) βρίσκονται και μετά γενικεύονται σε πιο αφηρημένες. Και μια μέση προσέγγιση (middle-out approach), στην οποία οι πιο σημαντικές έννοιες (concepts) εντοπίζονται και μετά γενικεύονται και εξειδικεύονται σε άλλες.

Οι **Gruninger και Fox** [18] προτείνουν μια μεθοδολογία που είναι εμπνευσμένη από την υλοποίηση ενός συστήματος γνώσης χρησιμοποιώντας λογική πρώτης τάξης (first order logic). Προτείνουν αρχικά να βρεθούν διαισθητικά τα βασικά σενάρια (πιθανές εφαρμογές στις οποίες πιθανότατα να χρησιμοποιηθεί η οντολογία). Αμέσως μετά ένα σύνολο ερωτήσεων φυσικής γλώσσας, που ονομάζονται ικανές ερωτήσεις (competency questions), χρησιμοποιούνται για να αποσαφηνιστεί το αντικείμενο της οντολογίας. Οι ερωτήσεις αυτές και οι απαντήσεις τους χρησιμοποιούνται για την εξαγωγή των βασικών εννοιών (concepts), των ιδιοτήτων τους (properties), των σχέσεων (relations) και των αξιωμάτων (axioms) της οντολογίας. Τα παραπάνω συστατικά της οντολογίας εκφράζονται σε λογική πρώτου βαθμού.

Στη μέθοδο που προτάθηκε στο **KACTUS** project [19] η οντολογία υλοποιείται με βάση μια εφαρμογή βάσης γνώσης (knowledge base KB) με αφηρημένο τρόπο (π.χ. στρατηγική bottom-up). Όσο η εφαρμογή υλοποιείται, τόσο πιο γενική γίνεται η οντολογία. Με άλλα λόγια οι συγγραφείς προτείνουν να ξεκινήσουμε την υλοποίηση μιας βάσης γνώσης (KB) για μια συγκεκριμένη εφαρμογή. Όταν μια νέα βάση γνώσης (KB) στο ίδιο πεδίο ενδιαφέροντος είναι απαραίτητη, προτείνουν να γενικεύσουμε την αρχική βάση γνώσης (KB) σε μια οντολογία και να την υιοθετήσουμε και για τις δυο εφαρμογές. Εφαρμόζοντας αυτή τη μέθοδο αναδρομικά, η οντολογία θα αναπαριστά τη συναινετική γνώση που χρειάζεται για όλες τις εφαρμογές. Ένας τρόπος για να εφαρμόσουμε αυτήν την προσέγγιση είναι να παράγουμε μια οντολογία με γενικευμένους όρους από πολλές βάσεις γνώσεις KBs που μοντελοποιούν το ίδιο πεδίο ενδιαφέροντος.

Η μέθοδος βασισμένη στο **Sensus** [20] είναι μια από πάνω προς τα κάτω προσέγγιση (top-down approach) για την παραγωγή συγκεκριμένων πεδίων ενδιαφέροντος οντολογιών από τεράστιες οντολογίες. Οι συγγραφείς προτείνουν την εύρεση όρων «σπόρους» (seed terms), που είναι σχετικοί με το συγκεκριμένο πεδίο. Οι όροι αυτοί συνδέονται σε μια ευρείας κάλυψης οντολογία (στην περίπτωση αυτή στην Sensus οντολογία που περιέχει περισσότερες από 50.000 έννοιες (concepts)). Όλες οι έννοιες στο μονοπάτι από τον «σπόρο» ως τη ρίζα περιλαμβάνονται στην οντολογία. Αν ένας όρος σχετικός με το πεδίο δεν εμφανίστηκε ακόμη τότε προστίθεται χειροκίνητα και αυτό συνεχίζεται μέχρι να μη λείπει κανείς σχετικός όρος. Τέλος, για τους κόμβους που έχουν πολλά μονοπάτια προς αυτούς, ολόκληρο το υποδέντρο κάτω από τον κόμβο μερικές φορές προστίθεται, και αυτό βασίζεται στην ιδέα, ότι

αν αρκετοί κόμβοι στο υπόδεντρο βρέθηκαν να είναι σχετικοί με το πεδίο, τότε και οι υπόλοιποι κόμβοι του υπόδεντρου θα είναι σχετικοί. Η προσέγγιση αυτή προάγει τον διαμοιρασμό (shareability) της γνώσης, αφού η ίδια βασική οντολογία χρησιμοποιείται για τη δημιουργία οντολογιών συγκεκριμένων πεδίων.

Η **METHODOLOGY** [21] είναι μια μεθοδολογία, που δημιουργήθηκε από το Ontological Engineering Group of the Technical University of Madrid (UPM), για την δημιουργία οντολογιών είτε από την αρχή, είτε επαναχρησιμοποιώντας άλλες οντολογίες όπως ακριβώς είναι, είτε μετά από τροποποίησή τους. Περιλαμβάνει: την εύρεση της διαδικασίας ανάπτυξής της οντολογίας, έναν κύκλο ζωής βασισμένο σε αυτόν που παρουσιάστηκε στο Σχήμα 2.2 και 2.3, και συγκεκριμένες τεχνικές για την εκτέλεση των διαφόρων δραστηριοτήτων.

Η **On-To-Knowledge** μεθοδολογία [22] προτείνει τα επόμενα βήματα για την υλοποίηση οντολογιών. Ορίζονται οι απαιτήσεις της οντολογίας, εντοπίζονται ικανές ερωτήσεις, μελετόνται πιθανές οντολογίες για επαναχρησιμοποίηση και υλοποιείται μια πρώτη έκδοση της οντολογίας. Αυτή βελτιώνεται, ώστε να προκύψει μια πιο ώριμη οντολογία που μπορεί να χρησιμοποιηθεί σε εφαρμογές. Γίνεται αξιολόγησή της, όπου ελέγχεται, αν οι απαιτήσεις της και οι ικανές ερωτήσεις καλύπτονται, και η οντολογία δοκιμάζεται σε κάποιο περιβάλλον μιας εφαρμογής. Τέλος, η τελική οντολογία μπαίνει στη διαδικασία της συντήρησης.

Μπορούμε τώρα να καταλήξουμε σε κάποια συμπεράσματα:

Καμία από τις προσεγγίσεις δεν καλύπτει όλες τις δραστηριότητες που εμπλέκονται στην ανάπτυξη οντολογιών. Οι περισσότερες από τις μεθόδους και τις μεθοδολογίες δίνουν μεγάλη έμφαση στις δραστηριότητες ανάπτυξης, ειδικά στη δραστηριότητα προσδιορισμού γνώσεων και εμπειριών (**conceptualization activity**) και στη δραστηριότητα υλοποίησης (**implementation activity**), και δε δίνουν τόση σημασία σε άλλες εξίσου σημαντικές, όπως τη διαχείριση (management), εκμάθηση (learning), συγχώνευση (merging), ενσωμάτωση (integration), εξέλιξη (evolution) και αξιολόγηση (evaluation). Αυτό γίνεται λόγω του ότι το πεδίο της μηχανικής των οντολογιών (ontological engineering) είναι σχετικά νέο.

Οι περισσότερες από τις προσεγγίσεις παρουσιάζουν κάποια μειονεκτήματα στη χρήση τους. Μερικές από αυτές δεν έχουν χρησιμοποιηθεί από άλλες ομάδες εκτός αυτών που τις δημιούργησαν, και σε μερικές περιπτώσεις έχουν χρησιμοποιηθεί σε απλά πεδία ενδιαφέροντος.

Οι περισσότερες από τις προσεγγίσεις δεν έχουν κάποιο συγκεκριμένο εργαλείο που να τους δίνει τεχνολογική υποστήριξη. Επίσης, κανένα από τα διαθέσιμα εργαλεία δεν καλύπτει όλες τις δραστηριότητες που είναι απαραίτητες για την ανάπτυξη οντολογιών.

2.7. Εργαλεία Οντολογιών

Λαμβάνοντας υπ όψιν τις πλατφόρμες λογισμικού που υποστηρίζουν τις περισσότερες δραστηριότητες του κύκλου ζωής ανάπτυξης οντολογιών, εδώ θα εστιάσουμε στα περιβάλλοντα εφαρμοσμένης μηχανικής οντολογιών νέας γενιάς (new generation of ontology engineering environments), και συγκεκριμένα στο Protégé, στο WebODE, στο OntoEdit και στο KAON. Τα περιβάλλοντα αυτά δημιουργήθηκαν για την ενσωμάτωση (integration) μιας οντολογίας σε πραγματικά συστήματα πληροφοριών και υποστηρίζουν τις περισσότερες δραστηριότητες του κύκλου ζωής μιας οντολογίας. Έχουν επεκτάσιμες, βασισμένες σε συστατικά αρχιτεκτονικές (component-based architectures), όπου νέα δομικά στοιχεία μπορούν εύκολα να προστεθούν για να προσφέρουν περισσότερη λειτουργικότητα στο εκάστοτε περιβάλλον.

Το **Protégé** [49] αναπτύχθηκε από το Stanford Medical Informatics (SMI) του Stanford University. Είναι ανοιχτού κώδικα (open source), μη-εξαρτημένη εφαρμογή (standalone application) με επεκτάσιμη αρχιτεκτονική (extensible architecture). Ο πυρήνας του περιβάλλοντος αυτού είναι ο συντάκτης οντολογιών (ontology editor) και περιέχει μια βιβλιοθήκη από πρόσθετα προγράμματα (plugins) που του δίνουν περισσότερη λειτουργικότητα.

Το **WebODE** [50] είναι η εξέλιξη του ODE (Ontology Design Environment), και αναπτύχθηκε στο UPM. Είναι επίσης ένα πακέτο για ανάπτυξη οντολογιών (ontology engineering suite) που δημιουργήθηκε με επεκτάσιμη αρχιτεκτονική. Το WebODE δεν είναι μια μη-εξαρτώμενη εφαρμογή (no standalone application), αλλά χρησιμοποιείται σαν ένας

διακομιστής (Web server). Ο πυρήνας του περιβάλλοντος αυτού είναι η υπηρεσία πρόσβασης οντολογιών (ontology access service), που χρησιμοποιείται από όλες τις υπηρεσίες και εφαρμογές που είναι συνδεδεμένες στον διακομιστή, ειδικά από τον WebODE συντάκτη οντολογιών (WebODE ontology editor). Οι οντολογίες του WebODE είναι αποθηκευμένες σε μια σχεσιακή βάση (relational database).

Το **OntoEdit** [51] αναπτύχθηκε από το AIFB ινστιτούτο του πανεπιστημίου Karlsruhe. Είναι όμοιο με τα προηγούμενα εργαλεία: επεκτάσιμο και ευέλικτο περιβάλλον, βασισμένο σε αρχιτεκτονική με επιπρόσθετα προγράμματα (plugins), που παρέχει λειτουργικότητα στην εξέταση και σύνταξη οντολογιών. Υπάρχουν δύο εκδώσεις του OntoEdit: το OntoEdit free και το OntoEdit Professional.

Το **KAON** πακέτο εργαλείων (tool suite) [52] είναι ένα ανοιχτού κώδικα (open source) επεκτάσιμο περιβάλλον για ανάπτυξη οντολογιών (ontology engineering environment). Ο πυρήνας αυτού του πακέτου είναι το ontology API, το οποίο ορίζει τη γνώση σε ένα μοντέλο που είναι βασισμένο σε μια επέκταση των RDF(S) (defines its underlying knowledge model based on an extension of RDF(S)). Το OI modeler είναι ένας συντάκτης οντολογιών που παρέχει δυνατότητες για την εξέλιξη της οντολογίας (ontology evolution), χαρτογράφηση της οντολογίας (ontology mapping), παραγωγή της οντολογίας από βάσεις δεδομένων (ontology generation from databases) κ.α.

Όταν κάποιος βρίσκεται στη φάση επιλογής ενός εργαλείου για την υλοποίηση μιας οντολογίας πρέπει να προσέξει κάποια από τα χαρακτηριστικά του. Έτσι, σημαντικό ρόλο παίζει η εκφραστικότητα του μοντέλου αναπαράστασης της γνώσης που υποστηρίζει το κάθε εργαλείο. Όλα τα εργαλεία που παρουσιάστηκαν παραπάνω επιτρέπουν την αναπαράσταση κλάσεων (classes), σχέσεων (relations), χαρακτηριστικών (attributes) και στιγμιότυπων (instances). Μόνο το KAON και το Protégé επιτρέπουν τη χρήση ποιο ευέλικτων συστατικών μοντελοποίησης όπως τις metaclasses. Επίσης, πριν την επιλογή κάποιου εργαλείου, κάποιος πρέπει να ελέγξει τις υπηρεσίες συμπερασμού (inference services) που υποστηρίζονται, όπως μηχανισμούς ελέγχου των περιορισμών (constraint) που μπορεί να υπάρχουν και συνέπειας, τύπους κληρονομικότητας, αυτόματη ταξινόμηση και χειρισμό εξαιρέσεων. Ένα ακόμη σημαντικό χαρακτηριστικό είναι η αρχιτεκτονική της εφαρμογής και η εξέλιξη των εργαλείων που υποστηρίζονται. Έτσι πρέπει να εξετάζονται ποιες πλατφόρμες

υλικού και λογισμικού είναι απαραίτητες για τη χρήση του εργαλείου, η αρχιτεκτονική του (standalone, client/server), αν είναι επεκτάσιμο, ο τρόπος που αποθηκεύουν τις οντολογίες (databases, ASCII files, etc.), η ανοχή σε σφάλματα (failure tolerance), η δημιουργία αντιγράφων ασφαλείας (backup management), η σταθερότητά τους (stability) και οι πολιτικές που ακολουθούνται για τη δημιουργία νέων εκδόσεων (versioning policies). Τέλος επίσης πολύ σημαντικό είναι να ελέγχεται η διαλειτουργικότητα (interoperability) με άλλα εργαλεία, συστήματα πληροφορίας και βάσεις δεδομένων, όπως και η δυνατότητα μεταφοράς από και προς κάποια γλώσσα οντολογιών.

2.8. Υλοποίηση Οντολογιών (Γλώσσες Οντολογιών)

Η δραστηριότητα υλοποίησης (που προτείνεται από όλες τις μεθοδολογίες και μεθόδους, καθώς υποστηρίζεται και από όλα τα εργαλεία ανάπτυξης) αποτελείται από την υλοποίηση μοντέλων κατανοητών από υπολογιστή (computable models) σε μια γλώσσα οντολογιών. Δύο είδη γλωσσών υλοποίησης οντολογιών υπάρχουν: οι κλασικές (classical) και οι markup. Παρακάτω θα περιγράψουμε τις πιο ουσιώδης.

Η **KIF** [53] είναι μια γλώσσα βασισμένη σε λογική πρώτης τάξης που δημιουργήθηκε σαν μια μορφή εμπλουτισμένου κειμένου (interchange format) για ποικίλα KR συστήματα. Η **ontolingua** [54] βασίζεται στην KIF και στη Frame Ontology [55]. Η KIF είναι η πιο εκφραστική από όλες τις γλώσσες που έχουν δημιουργηθεί για την αναπαράσταση οντολογιών, επιτρέποντας την αναπαράσταση των concepts, ταξινομιών (taxonomies) από concepts, n-διάστατες σχέσεις (n-ary relations), συναρτήσεις (functions), αξιώματα (axioms), στιγμιότυπα (instances) και διαδικασίες (procedures). Η υψηλή εκφραστικότητά της, οδήγησε σε δυσκολίες δημιουργίας μηχανισμών συμπερασμού (reasoning mechanisms) για αυτή.

Η **Loom** [56] δε δημιουργήθηκε αρχικά για την υλοποίηση οντολογιών, αλλά για γενικές knowledge bases (KBs). Η γλώσσα αυτή είναι βασισμένη στην περιγραφική λογική (description logics DL) και σε κανόνες παραγωγής (production rules), και παρέχει αυτόματη ταξινόμηση (automatic classification) των concepts. Η Loom επιτρέπει την αναπαράσταση των εξής συστατικών μιας οντολογίας: concepts, concepts taxonomies, n-ary relations, functions, axioms και production rules.

Η **FLogic** [57] (Frame Logic) συνδυάζει frames και λογική πρώτης τάξης, επιτρέποντας την αναπαράσταση των concepts, concept taxonomies, binary relations, functions, instances, axioms και deductive rules (συμπερασματικοί κανόνες). Η FLogic είναι η μοναδική από τις προηγούμενες γλώσσες που δεν έχουν Lisp-Like σύνταξη.

Το πρωτόκολλο **OKBC** [58] (Open Knowledge Base Connectivity) (που δεν θεωρείται ακριβώς γλώσσα) επιτρέπει πρόσβαση σε KBs αποθηκευμένες σε διαφορετικά συστήματα αναπαράστασης γνώσης. Από τα συστήματα που παρουσιάστηκαν παραπάνω, η Ontolingua και η Loom συμμορφώνονται με το OKBC.

Η **SHOE** [59] δημιουργήθηκε αρχικά σαν μια επέκταση της HTML και αργότερα σαν μια γλώσσα που χρησιμοποιεί τη σύνταξη της XML. Χρησιμοποιεί διαφορετικά tags από αυτά της HTML, που επιτρέπουν την εισαγωγή οντολογιών σε HTML κείμενα. Η SHOE συνδυάζει frames και rules. Επιτρέπει την αναπαράσταση concepts, concepts taxonomies, n-ary relations, instances και deduction rules, που χρησιμοποιούνται από τον μηχανισμό συμπερασμού (inference engine) για την απόκτηση νέας γνώσης.

Η **XOL** [60] δημιουργήθηκε στην προσπάθεια μετασχηματισμού ενός μικρού συνόλου του OKBC σε μια μορφή όμοια της XML (XMLization), που ονομάζεται OKBC-Lite. Είναι μια περιορισμένη γλώσσα στην οποία μπορούν να ορισθούν concepts, taxonomies και binary-relations. Δεν υπάρχει κάποιος μηχανισμός συμπερασμού που να την υποστηρίζει, αφού σχεδιάστηκε κυρίως για την ανταλλαγή οντολογιών στον τομέα της βιο-ιατρικής.

Η **RDF** [61] αναπτύχθηκε από την W3C (the World Wide Web Consortium) σαν μια γλώσσα βασισμένη στο σημασιολογικό δίκτυο (as a semantic-network based language) για την περιγραφή πόρων στο Web (Web resources). Τέλος, η **RDF Schema** [62] γλώσσα δημιουργήθηκε επίσης από την W3C σαν επέκταση στην RDF με αρχές βασισμένες σε frames (frame-based primitives). Ο συνδυασμός RDF και RDF Schema είναι γνωστός σαν RDF(S). Η RDF(S) είναι λιγότερη εκφραστική από τις προηγούμενες γλώσσες, επιτρέποντας την αναπαράσταση concepts, taxonomies και binary relations. Κάποιοι μηχανισμοί συμπερασμού (inference engines) δημιουργήθηκαν για τη γλώσσα αυτή, κυρίως για τον έλεγχο περιορισμών.

Οι γλώσσες αυτές δημιούργησαν την θεμελίωση του σημασιολογικού δικτύου (Semantic Web). Έτσι δημιουργήθηκαν τρεις ακόμη γλώσσες σαν προεκτάσεις της RDF(S): **OIL**, **DAML+OIL** και **OWL**.

Η σημασιολογική τυποποίηση (its formal semantics) της OIL [63] βασιζόταν στην περιγραφική λογική (description logics). Η DAML+OIL [64] επιτρέπει την αναπαράσταση των concepts, taxonomies, binary relations, functions και instances. Οι δύο αυτές γλώσσες δεν χρησιμοποιούνται πλέον.

Τέλος, το 2001 η W3C δημιούργησε μια ομάδα εργασίας που ονομαζόταν Web Ontology (WebOnt) Working Group. Στόχος της ομάδας αυτής ήταν η δημιουργία μιας νέας markup γλώσσας για το σημασιολογικό Web (Semantic Web), με το όνομα **OWL** [65] (Ontology Web Language). Η γλώσσα αυτή συστάθηκε από την W3C τον Φεβρουάριο του 2004.

2.9. Είδη Οντολογιών

Πολλοί συγγραφείς βλέπουν τις οντολογίες από διαφορετικές οπτικές γωνίες. Έτσι, δε μας προκαλεί έκπληξη το γεγονός ότι στη βιβλιογραφία βρίσκουμε ποικίλες ταξινομήσεις των οντολογιών, που η κάθε μια επικεντρώνεται σε διαφορετικά κριτήρια.

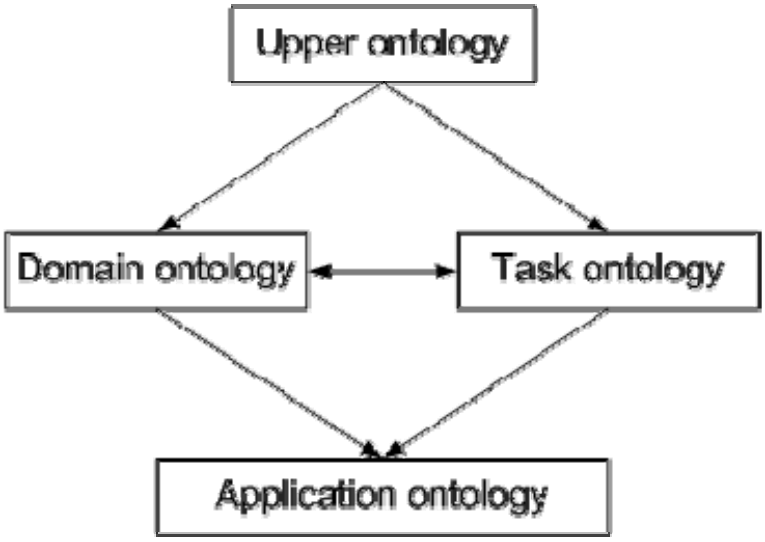
Σύμφωνα με το επίπεδο γενικότητας (generality level) που παρουσιάζεται στο Σχήμα 2.4., ο Guarino θεώρησε τους εξής τύπους οντολογιών [66]:

High-level ontologies: που περιγράφουν γενικές έννοιες (concepts) όπως χώρος (space), χρόνος (time), ύλη (material). Δεν εξαρτώνται από κάποιο συγκεκριμένο πεδίο ενδιαφέροντος ή κάποιο συγκεκριμένο πρόβλημα. Σκοπός τους είναι η ενοποίηση κριτηρίων που μεγάλες κοινότητες χρηστών τα αντιλαμβάνονται με διάφορους τρόπους.

Domain ontologies: περιγράφουν το λεξιλόγιο που είναι σχετικό με ένα γενικό πεδίο (πχ. σε συστήματα πληροφορίας ή στην ιατρική), με εξειδίκευση των εννοιών (concepts) που εισήχθησαν στις υψηλού επιπέδου οντολογίες (high-level ontologies).

Task ontologies: περιγράφουν το λεξιλόγιο που σχετίζεται με γενικές εργασίες ή δραστηριότητες (πχ. η φάση ανάπτυξης ή οι πωλήσεις), με εξειδίκευση των εννοιών (concepts) που εισήχθησαν στις υψηλού επιπέδου οντολογίες (high-level ontologies).

Application ontologies: περιγράφουν έννοιες που ανήκουν ταυτόχρονα σε ένα πεδίο και σε μια συγκεκριμένη εργασία, με εξειδίκευση των εννοιών (concepts) των domain ontologies και task ontologies. Συνήθως ανταποκρίνονται σε ρόλους που παίζουν οι οντότητες ενός πεδίου όταν εκτελούν κάποια δραστηριότητα, δηλαδή οι οντολογίες αυτού του είδους είναι αφοσιωμένες σε μια συγκεκριμένη εφαρμογή.



Σχήμα 2.4 Είδη Οντολογιών σύμφωνα με τον Guarino. [66]

Από την άλλη ο Fensel [67], συνέστησε την εξής εναλλακτική ταξινόμηση:

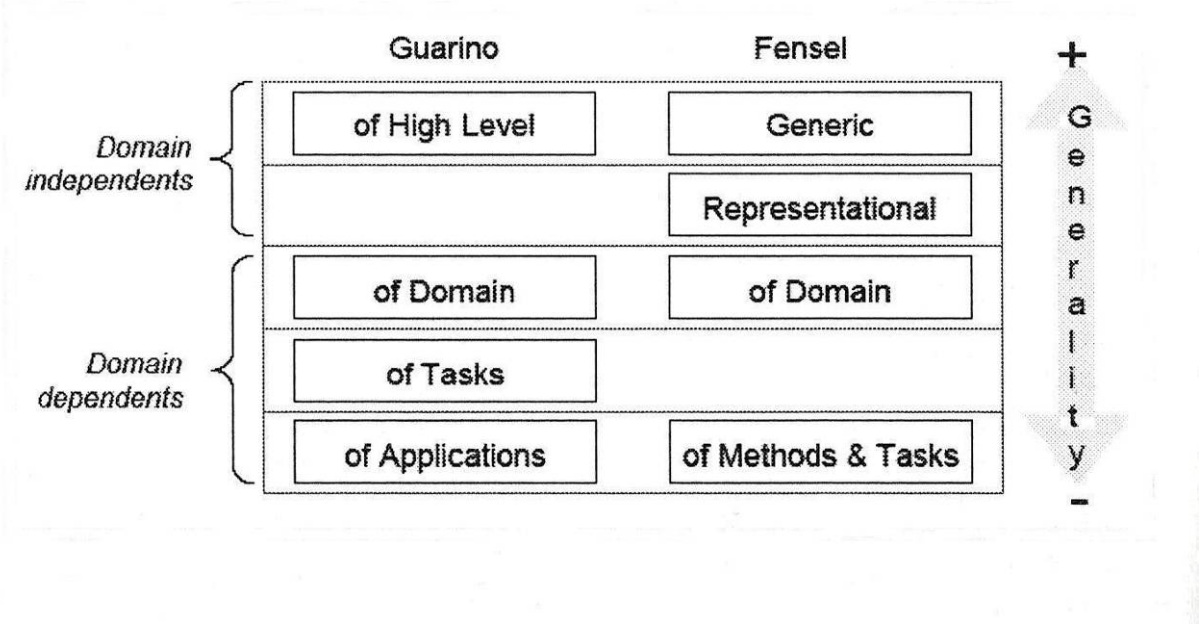
Generic or common-sense ontologies: που λαμβάνουν υπ όψιν τη γενική γνώση που υπάρχει στον κόσμο. Παρέχουν βασικές αντιλήψεις και έννοιες (concepts) για τον χώρο (space), τον χρόνο (time), την κατάσταση (state), των συμβάντων (events), που είναι έγκυρες για ποικίλα πεδία.

Representational ontologies: που δεν ανήκουν σε κανένα συγκεκριμένο πεδίο. Προσφέρουν οντότητες (entities) χωρίς να συνιστούν τι θα μπορούσαν να αναπαραστήσουν. Έτσι, ορίζουν έννοιες (concepts) που εκφράζουν γνώση για μια object-oriented ή framework-oriented προσέγγιση.

Domain ontologies: που αποτυπώνουν γνώση έγκυρη για ένα συγκεκριμένο τύπο πεδίου ενδιαφέροντος (π.χ. για την ηλεκτρονική, για την ιατρική, κ.α.).

Method and task ontologies: οι πρώτες προσφέρουν ορολογία για συγκεκριμένες μεθόδους ανάλυσης ενός προβλήματος, ενώ οι δεύτερες παρέχουν όρους για συγκεκριμένες εργασίες. Και οι δύο προσφέρουν μια λογική άποψη για τη γνώση που είναι συγκεντρωμένη σε ένα πεδίο ενδιαφέροντος.

Μια αντιστοίχηση των δύο προηγούμενων ταξινομήσεων φαίνεται στο Σχήμα 2.5.



Σχήμα 2.5 Είδη Οντολογιών Σύμφωνα με το Επίπεδο Γενικότητας. [86]

Ένας άλλος πιθανός τρόπος ταξινόμησης των οντολογιών είναι με βάση τη φύση του πραγματικού κόσμου (according to the nature of the real-world issue) που πρόκειται να

μοντελοποιηθεί. Με βάση αυτή τη λογική οι Jurisica et al. όρισαν την επόμενη ταξινόμηση [68]:

Static ontologies: περιγράφουν πράγματα τα οποία υπάρχουν, τα χαρακτηριστικά τους και τις σχέσεις που υπάρχουν μεταξύ τους. Η κατηγορία αυτή υποθέτει πως ο κόσμος αποτελείται από οντότητες που είναι προικισμένες με μία μοναδική και αμετάβλητη ταυτότητα. Σε αυτές χρησιμοποιούμε όρους όπως οντότητα (entity), χαρακτηριστικό (attribute) και σχέση (relationship).

Dynamic ontologies: που περιγράφουν τις όψεις του μοντελοποιημένου κόσμου που μπορεί να αλλάζουν με το χρόνο. Για τη μοντελοποίηση κάτι τέτοιου ίσως να είναι απαραίτητη η χρήση μηχανών πεπερασμένων καταστάσεων (finite state machines), δίκτυα Petri (Petri nets), κ.α. Η επεξεργασία (process), η κατάσταση (state), η μετάβαση κατάστασης (state transition) είναι παραδείγματα ορολογίας που συνήθως περιλαμβάνονται στην κατηγορία αυτή.

Intentional ontologies: που περιγράφουν τα κίνητρα (motivations), τους σκοπούς (intentions), τους στόχους (goals), τις αρχές (beliefs), τις εναλλακτικές (alternatives) και τις προτιμήσεις (elections) των εμπλεκόμενων συντελεστών (involved agents). Κάποιοι τυπικοί όροι σε τέτοιου είδους οντολογίες είναι οι όψεις (aspects), το αντικείμενο (object), οι συντελεστές (agents) και η υποστήριξη (support).

Social ontologies: που περιγράφουν κοινωνικές όψεις όπως δομές οργάνωσης (organizational structures), δίκτυα (nets) ή αλληλεξαρτήσεις (interdependences). Για το λόγο αυτό περιλαμβάνουν όρους όπως δράστης (actor), θέση (position), ρόλος (role), αρχή (authority), ευθύνη (responsibility) ή δέσμευση (commitment).

Μερικοί συγγραφείς πιστεύουν πως αυτός ο γραμμικός τρόπος ταξινόμησης των οντολογιών, που βασίζεται σε ένα και μοναδικό κριτήριο δεν επιτρέπει την επαρκή απόδοση της πολυπλοκότητας των προβλημάτων. Έτσι οι Gomez-Perez et al. [69] προτείνουν μια δισδιάστατη ταξινόμηση, λαμβάνοντας υπ όψιν δύο κριτήρια: το πόσο πλούσια είναι η εσωτερική δομή, και το θέμα με το οποίο ασχολείται ο conceptualization.

Σε αυτή τη δισδιάστατη πρόταση για ταξινόμηση, κάθε οντολογία ανήκει σε μια από τις παρακάτω κατηγορίες με βάση το πόσο πλούσια είναι η εσωτερική της δομή (richness of its internal structure):

Controlled vocabularies: που αποτελούνται από μια πεπερασμένη λίστα όρων. Π.χ. θα μπορούσε να χαρακτηριστεί έτσι μια λίστα που περιλαμβάνει τις υπηρεσίες και τα προϊόντα ενός οργανισμού.

Glossaries: που είναι λίστες όρων με τους ορισμούς τους σε φυσική γλώσσα. Η δομή των glossaries είναι όπως αυτή ενός λεξικού, που περιλαμβάνει τους όρους ταξινομημένους σε αλφαβητική σειρά, και παραθέτει τους ορισμούς τους.

Thesauruses: Είναι λίστες όρων που χρησιμοποιούνται κυρίως για δεικτοδότηση (for indexing purposes). Διαφοροποιούνται από την προηγούμενη κατηγορία στο ότι εκτός από τους ορισμούς των όρων, περιλαμβάνουν και σημασιολογικές σχέσεις μεταξύ των όρων, όπως σχέσεις ιεραρχίας "γονέα-παιδιού" (π.χ. whole-part, genus-species, type-instance) και σχέσεις συσχέτισης (π.χ. related terms).

Informal is-a hierarchies: που περιλαμβάνουν ιεραρχίες όρων που δεν ακολουθούν αυστηρά τη σχέση γενίκευσης "is-a". Για παράδειγμα οι όροι "rental vehicle" και "hotel" μπορούν να μοντελοποιηθούν άτυπα (informally) κάτω από την όρο "travel" αφού θεωρούνται κύριοι συστατικά του τομέα "traveling", αλλά δεν αποτελούν μια γενίκευση του όρου αυτού, αφού δεν μπορούμε να πούμε πως είναι κάποιοι τύποι ταξιδιού "types of traveling".

Formal hierarchies: στην περίπτωση αυτή, υπάρχει μια αυστηρή "is-a" σχέση γενίκευσης μεταξύ των στιγμιότυπων μιας κλάσης (class) και της αντίστοιχης υπερκλάσης (superclass). Για παράδειγμα ένας δάσκαλος (a teacher) "is-a" άνθρωπος (people). Στόχος εδώ είναι να αξιοποιηθεί η κληρονομικότητα (inheritance).

Frames: που είναι ένα μοντέλο που περιλαμβάνει σαν πρωταρχικά συστατικά του, classes ή frames, που έχουν properties, slots ή attributes. Παρέχουν ένα τρόπο μοντελοποίησης ενός πεδίου ενδιαφέροντος. Ένα frame μπορεί να περιέχει ως τιμή ενός slot, μια που να αναφέρεται σε κάποιο άλλο frame, έτσι μοντελοποιούνται και οι σχέσεις (relations) μεταξύ

των frames. Έτσι περιέχουν κλάσεις (classes) σαν ιδιότητες, που μπορούν να κληρονομηθούν από άλλες κλάσεις σε χαμηλότερα επίπεδα σε μια τυπική (formal) "is-a" ταξινομία (taxonomy).

Ontologies with value constraints: που μπορούν να εκφράσουν περιορισμούς στις τιμές. Η πιο τυπική περίπτωση είναι αυτή της επιβολής περιορισμών που εξαρτώνται στον τύπο δεδομένων μιας ιδιότητας (πχ. μια μέρα του μηνός πρέπει να είναι μικρότερη του 32).

Ontologies with generic logical constraints: αυτές είναι οι πιο εκφραστικές οντολογίες που επιτρέπουν την αναπαράσταση συγκεκριμένων περιορισμών μεταξύ των όρων της οντολογίας χρησιμοποιώντας λογική πρώτης τάξης (first-order logic).

Ταυτόχρονα σχετικά με το θέμα του conceptualization (subject of the conceptualization), μια οντολογία θα ανήκει σε έναν από τους παρακάτω τύπους:

Knowledge representation ontologies: που παρέχουν πρωταρχικά συστατικά μοντελοποίησης, μοντέλων αναπαράστασης γνώσης. Προσδίδουν τα συστατικά μοντελοποίησης που χρησιμοποιούνται σε frame-based αναπαραστάσεις, όπως κλάσεις (classes), υποκλάσεις (subclasses), τιμές (values), ιδιότητες (attributes ή slots) και αξιώματα (axioms).

High-Level or Upper-Level or Top-Level ontologies: που περιγράφουν πολύ γενικές έννοιες (concepts) και αντιλήψεις (notions) που μπορούν να σχετίζονται με τους όρους της ρίζας (root terms) όλων των οντολογιών. Οι οντολογίες αυτές μπορεί να σχετίζονται με τον χρόνο (time), τον χώρο (space), συμβάντα (events), όρια (boundaries), εμπλεκόμενους (agents), ρόλους (roles) κ.α.. Ένα πρόβλημα που παραμένει άλυτο είναι ότι πολλές από τις high-level οντολογίες διαφέρουν στον τρόπο που ταξινομούν τα concepts τους. Αυτό επιφέρει δυσκολίες στην ενσωμάτωση (integrate) και στην ανταλλαγή (exchange) των οντολογιών.

Domain ontologies: που περιέχουν οντολογίες που μπορούν να επαναχρησιμοποιηθούν σε κάποιο συγκεκριμένο πεδίο (π.χ. ιατρική, εφαρμοσμένη μηχανική (engineering), κ.α.).

Παρέχουν ένα λεξικό για concepts σχετικά με το πεδίο και τις σχέσεις μεταξύ τους (relationships).

Task ontologies: περιγράφουν το λεξικό που είναι σχετικό με κάποια γενική δραστηριότητα ή συγκεκριμένη εργασία. Παρέχουν ένα λεξικό με όρους που χρησιμοποιούνται για την επίλυση προβλημάτων που μπορεί αλλά μπορεί και όχι να ανήκουν στο ίδιο πεδίο.

Domain task ontologies: διαφοροποιούνται από την προηγούμενη κατηγορία στο ότι αυτές είναι επαναχρησιμοποιήσιμες σε ένα δοθέντα πεδίο και όχι μεταξύ διαφορετικών πεδίων.

Method ontologies: που παρέχουν ορισμούς σχετικών εννοιών (concepts) και των σχέσεών τους. Είναι εφαρμόσιμες σε μια διαδικασία αιτιολόγησης (reasoning process) που είναι ειδικά σχεδιασμένη για να φέρει εις πέρας μια συγκεκριμένη εργασία (task).

Application ontologies: που εξαρτώνται από την εφαρμογή. Συνήθως επεκτείνουν και εξειδικεύουν το λεξιλόγιο μιας domain ontology ή μιας task ontology για κάποια συγκεκριμένη εφαρμογή.

Στη βιβλιογραφία των οντολογιών, χρησιμοποιούνται επίσης τα επίθετα “formal”, “informal” και “semi-formal”. Στην περίπτωση αυτή σε ποια από τις τρεις παραπάνω κατηγορίες ανήκει μια οντολογία, παίζει ρόλο το πόσο τυπική (formal) ή όχι είναι η γλώσσα με την οποία θα αναπαρασταθεί η οντολογία. Έτσι οι οντολογίες που εκφράζονται χρησιμοποιώντας φυσική γλώσσα θεωρούνται πλήρως “informal”, και αυτές που αναπαριστούνται χρησιμοποιώντας πρώτης τάξης λογική (first order logic) θεωρούνται “formal” [70]. Μια ενδιάμεση κατάσταση υπάρχει όταν η οντολογία αναπαρασταθεί με χρήση των UML class diagrams, οπότε θεωρείται πως είναι “semi-formal”.

Οι Uschold και Jasper [70] με σκοπό να δώσουν έναν απλό ορισμό που θα εξυπηρετούσε τα διάφορα πεδία εφαρμογών (knowledge engineering, databases, software engineering, κ.α.), και θα ήταν κατανοητός και από μη ειδικούς, εξέφρασαν τον εξής χαρακτηρισμό:

“Μία οντολογία μπορεί να πάρει ποικίλες μορφές, αλλά απαραίτητα θα εμπεριέχει ένα λεξικό όρων, και κάποιον προσδιορισμό της σημασίας τους. Αυτό περιλαμβάνει ορισμούς και μια

ένδειξη για το πώς οι έννοιες (concepts) αλληλεξαρτώνται, τα οποία συλλογικά επιβάλουν μια δομή στο πεδίο και περιορίζουν τις πιθανές παρερμηνείες των όρων.”

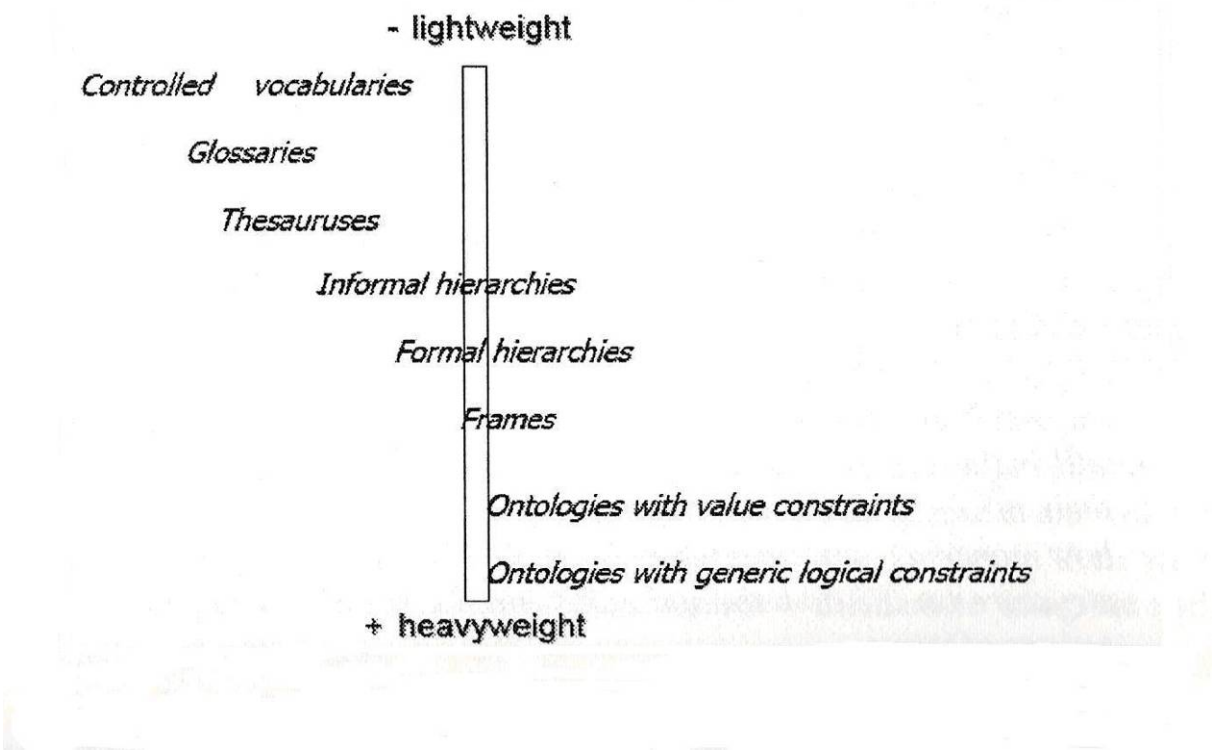
Με την ίδια επιδίωξη οι Gomez-Perez et al. [69] εξέφρασαν το εξής:

“Μια οντολογία στοχεύει στην απόδοση της συναινετικής γνώσης με ένα γενικό τρόπο, και θα μπορεί να επαναχρησιμοποιηθεί και διαμοιραστεί μεταξύ των διάφορων εφαρμογών λογισμικού και μεταξύ ομάδων (groups) και ατόμων.”

Heavyweight έναντι Lightweight οντολογίες

Συχνά στην κοινότητα των ανθρώπων που ασχολούνται με οντολογίες, ακούγονται οι έννοιες lightweight και heavyweight οντολογίες. Ο διαχωρισμός αυτός είναι μια απλοποίηση της ταξινόμησης των οντολογιών με βάση το πόσο πλούσια είναι η εσωτερική τους δομή (richness of its internal structure). Έτσι οι lightweight οντολογίες είναι κυρίως ταξινομίες, ενώ οι heavyweight είναι αυτές που μοντελοποιούν κάποια συγκεκριμένη γνώση “σε μεγαλύτερο βάθος και παρέχουν περισσότερους περιορισμούς στην σημασιολογία του συγκεκριμένου πεδίου” [69]. Η πρώτη κατηγορία περιλαμβάνει concepts, concept taxonomies, relationships μεταξύ των concepts, και ιδιότητες που περιγράφουν τα concepts. Η δεύτερη προσθέτει αξιώματα και περιορισμούς με σκοπό να αποσαφηνιστεί το νόημα των όρων. Στο Σχήμα 2.6 βλέπουμε μια ταξινόμηση των κατηγοριών που παρατέθηκαν στην προηγούμενη ενότητα (όσον αφορά το πόσο πλούσια είναι η εσωτερική τους δομή) ανάλογα με το κατά πόσο πλησιάζουν στις lightweight ή heavyweight οντολογίες

.



Σχήμα 2.6 Μια ταξινόμηση των Οντολογιών από Lightweight σε Heavyweight. [69]

ΚΕΦΑΛΑΙΟ 3. ΟΙ ΟΝΤΟΛΟΓΙΕΣ ΣΤΗΝ ΙΑΤΡΙΚΗ

3.1 Οντολογίες στην Ιατρική και Βιοϊατρική (OpenCyc, WordNet, OpenGalen, UMLS, SNOMED CT, FMA)

3.2. Αναπαράσταση του Ιατρικού Πεδίου σε Γενικές Οντολογίες

3.3. Παραδείγματα Ιατρικών Οντολογιών

Σήμερα όπως προείπαμε, η ιατρική οντολογία είναι η ραχοκοκαλιά αξιόπιστων και αποτελεσματικών εφαρμογών στον τομέα της ιατρικής φροντίδας. Μπορεί να βοηθήσει στην ανάπτυξη πιο ισχυρών και διαλειτουργικών συστημάτων πληροφορίας στον τομέα αυτό. Μπορούν να υποστηρίξουν την ανάγκη του τομέα της ιατρικής φροντίδας στην μεταφορά, επαναχρησιμοποίηση και διαμοιρασμό των δεδομένων των ασθενών, και παρέχουν κριτήρια βασισμένα σε σημασιολογία, για να υποστηριχτούν στατιστικά στοιχεία για διάφορους σκοπούς.

Η σχεδίαση και υλοποίηση οντολογιών στην ιατρική εστιάζει κυρίως στην αναπαράσταση και οργάνωση των ιατρικών ορολογιών. Οι γιατροί αναπτύσσουν τις δικές τους γλώσσες και λεξικά για να διατηρήσουν και να ανταλλάσσουν ιατρικές γνώσεις και πληροφορίες σχετικές με τους ασθενείς, με αποτελεσματικό τρόπο. Αυτές οι ορολογίες περιέχουν μεγάλες ποσότητες εννοούμενης πληροφορίας. Από την άλλη μεριά τα συστήματα ιατρικής πληροφορίας, χρειάζεται να είναι σε θέση να επικοινωνούν ξεκάθαρα με σύνθετες και αναλυτικές ιατρικές έννοιες (που μπορεί να είναι εκφρασμένες και σε διαφορετικές γλώσσες). Αυτό είναι φανερά μια δύσκολη διαδικασία, και χρειάζεται μια βαθιά ανάλυση της δομής και των εννοιών της ιατρικής ορολογίας, προκειμένου να καθοριστούν domain ontologies ικανές να παρέχουν ευκαμψία και συνέπεια σε συστήματα ιατρικής πληροφορίας.

Λόγω του ότι η παρούσα εργασία, είναι μεν επάνω στις οντολογίες αλλά επικεντρώνεται στη δημιουργία μιας domain ontology, με πεδίο ενδιαφέροντος τις καρδιαγγειακές παθήσεις, θεωρήθηκε αναγκαίο να παρουσιαστεί η σχετική δουλειά που έχει πραγματοποιηθεί στο

πεδίο των οντολογιών στην ιατρική. Έτσι στη συνέχεια της ενότητας αυτής περιγράφονται σχετικές εργασίες που έχουν πραγματοποιηθεί στο πεδίο της ιατρικής.

Τόσο η υψηλή χρησιμότητα των ιατρικών οντολογιών, όσο και διάφορες εργασίες που έχουν γίνει για μεθοδολογικά θέματα πάνω στις οντολογίες αλλά και εργασίες που έχουν πιο εφαρμόσιμο χαρακτήρα, παρουσιάζονται στην εργασία των Francesco Pinciroli και Domenico M. Pisanelli [1].

3.1. Οντολογίες στην Ιατρική και Βιοϊατρική (OpenCyc, WordNet, OpenGalen, UMLS, SNOMED CT, FMA)

Σκοπός των ιατρικών οντολογιών είναι η μελέτη κλάσεων οντοτήτων, που έχουν ιατρικό ενδιαφέρον. Παραδείγματα τέτοιων κλάσεων περιλαμβάνουν συστατικά όπως η μιτροειδής βαλβίδα (mitral valve) και η γλυκόζη (glucose), ποιοτικά χαρακτηριστικά όπως η διάμετρος της αριστερής κοιλίας (diameter of the left ventricle) και η καταλυτική δράση των ενζύμων (catalytic function of enzymes), και διαδικασίες όπως η κυκλοφορία του αίματος (blood circulation) και η έκκριση ορμονών. Αντίθετα από την ιατρική ορολογία (medical terminology), της οποίας ο σκοπός είναι η συλλογή των ονομάτων από οντότητες που χρησιμοποιούνται στον ιατρικό τομέα, η ιατρική οντολογία ενδιαφέρεται για τον ορισμό των ιατρικών κλάσεων και τις σχέσεις που υπάρχουν μεταξύ τους.

Όπως προαναφέραμε, οι οντολογίες μπορούν να κατηγοριοποιηθούν ανάλογα με το πεδίο που προσπαθούν να περιγράψουν, ή το επίπεδο αναλυτικότητας που παρέχουν, και στην κατηγοριοποίηση σύμφωνα με το πεδίο που αναπαριστούν περιλαμβάνονται οι upper και domain οντολογίες. Οι κύριες κατηγορίες που αναπαρίστανται στις οντολογίες θα πρέπει να είναι διαμοιραζόμενες μεταξύ των οντολογιών. Έτσι οι κατηγορίες των upper οντολογιών θα πρέπει να είναι συμβατές με τις ισοδύναμες σημασιολογικά περιοχές στις αντίστοιχες domain οντολογίες. Π.χ. η έννοια “Ασθένεια” σε μία upper οντολογία θα πρέπει να είναι συμβατή με την ισοδύναμη έννοια σε μια ιατρική domain οντολογία. Επιπλέον οι γενικές θεωρίες θα πρέπει να είναι επίσης διαμοιραζόμενες σε κάθε οντολογία. Έτσι πχ. μια αναπαράσταση της ανατομίας θα έπρεπε να επαναχρησιμοποιεί μια γενική θεωρία του χώρου (a generic theory of spatial objects). Με τη σειρά της, αφού η ανατομία παίζει κεντρικό ρόλο στην βιοιατρική και

είναι ιδιαίτερα σταθερή, μια οντολογία της ανατομίας μπορεί να εξυπηρετήσει σαν αναφορά, για οντολογίες που βασίζονται στην αναπαράσταση του ανθρώπινου σώματος, πχ. μια οντολογία για ασθένειες. Στην πράξη όμως αυτό είναι ιδανικό και δεν επιτυγχάνεται πάντα. Η κατασκευή ιατρικών οντολογιών που παρέχουν διαμοιραζόμενη γνώση σε ανθρώπους και υπολογιστές είναι μια πρόκληση.

Οι οντολογίες παίζουν πρωταρχικό ρόλο στην έρευνα της ιατρικής πληροφορικής, συμβάλλοντας για παράδειγμα στην επεξεργασία φυσικής γλώσσας (πχ. **Hahn et al. 1999)**, στη διαλειτουργικότητα μεταξύ συστημάτων (π.χ. **Degoulet et al. 1998**), και πρόσβαση σε ανομοιογενείς πηγές πληροφορίας, περιλαμβάνοντας και το σημασιολογικό δίκτυο (πχ. **Pisanelli et al. 2004**). Οι οντολογίες όλο και πιο πολύ, συμμετέχουν στη χρήση διάφορων πηγών πληροφορίας σε μια ποικιλία εφαρμογών.

Παρακάτω δίνεται μια περιγραφή κάποιων σημαντικών υπαρκτών οντολογιών για το πεδίο της ιατρικής, με κάποια χαρακτηριστικά τους. Αρχικά θα εξεταστεί πως αναπαρίστανται περιοχές της ιατρικής σε κάποια γενικά συστήματα, όπως το OpenCyc και το WordNet. Μετέπειτα περιγράφονται τρία συστήματα του ιατρικού πεδίου, GALEN, UMLS και SNOMED CT. Επίσης εξετάζεται μια αναφορική οντολογία, η Foundational Model of Anatomy (FMA) [71].

3.2. Αναπαράσταση του Ιατρικού Πεδίου σε Γενικές Οντολογίες

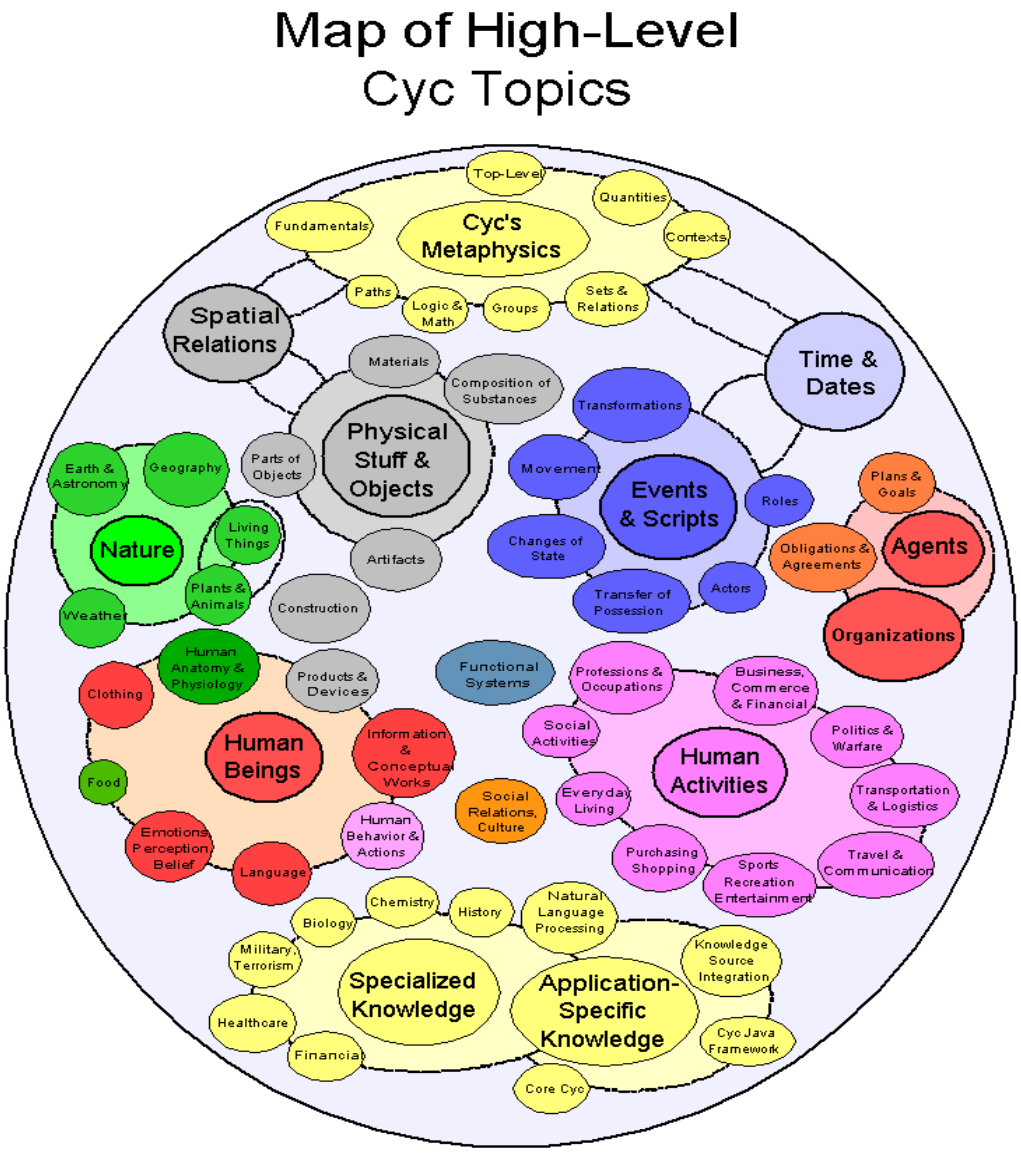
3.2.1. OpenCyc

Η Cyc είναι μια γενική οντολογία που αναπτύχθηκε από την Cycorp, Inc. Είναι βασισμένη πάνω σε έναν πυρήνα περισσότερων από 1.000.000 ισχυρισμών (assertions) που είναι εκφρασμένοι στη γλώσσα CycL και συλλαμβάνουν "κοινής αίσθησης" γνώση και βοηθούν στη δημιουργία μιας ποικιλίας εφαρμογών που ασχολούνται εντατικά με τη γνώση (knowledge-intensive applications). Περιλαμβάνει κάποιες "μικροθεωρίες" (microtheories) που είναι σύνολα από ισχυρισμούς (assertions) που διαμοιράζονται ένα κοινό σύνολο από υποθέσεις (assumptions) και έχουν ως στόχο να συσχετίσουν όρους μεταξύ τους και να σχηματίσουν μια upper οντολογία που να αφορά οτιδήποτε γύρω από την κοινή

πραγματικότητα των ανθρώπων. Η OpenCyc, το υψηλό της επίπεδο, είναι το δημόσια διαθέσιμο κομμάτι της οντολογίας, και περιέχει 6.000 έννοιες και 60.000 ισχυρισμούς για τις έννοιες αυτές. Τα ονόματα των εννοιών (concepts names) στην Cyc ονομάζονται "σταθερές" (constants), αρχίζουν με ένα "#$" και υποστηρίζονται οι εξής
:
Τα "'άτομα" (individuals) όπως "#$BillClinton και "#$France
Τις "συλλογές" (collections) όπως "#$Tree-ThePlant" που περιλαμβάνει όλα τα δέντρα. Ένα μέλος μιας συλλογής ονομάζεται στιγμιότυπο της συλλογής αυτής (instance of the collection)
Τις "συναρτήσεις" (functions) που παράγουν όρους για έναν δοσμένο όρο, π.χ. η "#$FruitFn" όταν παρέχεται με όρισμα μια συλλογή από φυτά, θα επιστρέψει τη συλλογή των φρούτων τους.
Οι δύο πιο σημαντικές σχέσεις στην OpenCyc είναι οι εξής:
Η "#$isa" που περιγράφει πως ένα άτομο είναι στιγμιότυπο μιας συλλογής
Η "#$genls" που περιγράφει πως μια συλλογή είναι υποκατηγορία μιας άλλης

Κάποια παραδείγματα στον τομέα της ιατρικής είναι τα εξής:
Ο "καρκίνος" (cancer) είναι στιγμιότυπο του τύπου "τύπος ασθένειας" (disease type) (#$isa #$Cancer #$DiseaseType),
Ο "καρκίνος" είναι υποκατηγορία του "κατάσταση ασθένειας" (#$genls #$Cancer #$AilmentCondition)
Η συνάρτηση "#$CancerFn" εκφράζει πως κάποιο μέρος του σώματος μπορεί να είναι σημείο εμφάνισης καρκίνου. Της δίνουμε σαν όρισμα ένα μέρος τους σώματος και επιστρέφει συγκεκριμένους καρκίνους (#$CancerFn #$Throat)

Στο Σχήμα 3.1. δίνεται μια αναπαράσταση των θεμάτων υψηλού επιπέδου, την γνώση για τα οποία αποδίδει η OpenCyc.



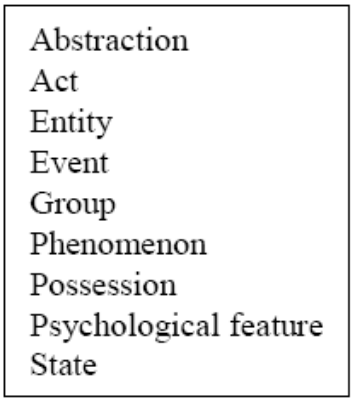
Σχήμα 3.1 Τα Θέματα Υψηλού Επιπέδου που Αποδίδει η OpnCyc. [71].

3.2.2. WordNet

Το WordNet είναι μια ηλεκτρονική λεξική βάση που αναπτύχθηκε στο Princeton University, και εξυπηρετεί σαν πηγή για εφαρμογές που σχετίζονται με την επεξεργασία της φυσικής γλώσσας (Natural language processing NLP) και την ανάκτηση πληροφορίας (Information retrieval). Η κεντρική δομή στο WordNet είναι ένα σύνολο συνωνύμων (synset) που αναπαριστούν μια έννοια (concept). Ο σχηματισμός των synset βασίζεται στην συνωνυμία (synonymy), το ότι δηλαδή η απόδοση μιας ερμηνείας εκφράζεται από πολλούς όρους και στην πολυσημία (polysemy), δηλαδή πως μια λέξη μπορεί να έχει διαφορετικές ερμηνείες.

Περιέχει διαφορετικές δομές για κάθε γλωσσική κατηγορία που καλύπτει, ουσιαστικά, ρήματα, επίθετα και επιρρήματα. Έτσι π.χ. το επίθετο "νεφρικός" (renal) και το ουσιαστικό "νεφρό" (kidney), αν και είναι όμοια σε νόημα, ανήκουν σε διαφορετικές δομές, και μια σχέση "σχετικότητας" (pertainymy), συσχετίζει τους δύο τύπους.

Η τρέχουσα έκδοση του WordNet περιέχει περισσότερα από 117.000 synset ουσιαστικών, που οργανώνονται σε εννέα ιεραρχίες, η κάθε μια από τις οποίες ξεκινά με ένα "μοναδικό αρχικό synset" (unique beginner). Στο Σχήμα 3.2 βλέπουμε το υψηλό επίπεδο της κάθε ιεραρχίας.



Σχήμα 3.2 Το Υψηλό Επίπεδο του WordNet. [71]

Κάθε synset στην ιεραρχία ουσιαστικών συμμετέχει τουλάχιστον σε μια is-a σχέση (hyponymy, η σημασιολογία της λέξης εμπεριέχεται σε αυτή μιας άλλης), και μπορεί επιπρόσθετα να συμμετέχει σε πολλές part-of-like σχέσεις (meronymy, μία έννοια αποτελεί μέλος μιας άλλης, π.χ. το δάχτυλο είναι μέλος του χεριού). Οι σχέσεις υπονυμίας (hyponymy), δημιουργούνται μεταξύ των synsets ακολουθώντας των εξής ορισμό: Μια έννοια που αποδίδεται από το synset {x, x', …} λέγεται υπονύμιο της έννοιας που αποδίδεται από το synset {y, y', …} αν οι ομιλούντες της αγγλικής γλώσσας δέχονται ότι το x είναι ένα είδος του y.

Πολλές έννοιες που αποδίδουν διαταραχές στην υγεία, σε ιατρικές ορολογίες, όταν εντοπίζονται στο WordNet, κατηγοριοποιούνται καταλλήλως. Πχ. Η λευχαιμία (leukemia) είναι υπώνυμο (hyponym) του καρκίνου (cancer). Ωστόσο, σε μερικές περιπτώσεις ένα

ιατρικό σημάδι ή σύμπτωμα, εμφανίζεται σαν υπώνυμο (hyponym) μιας μη ιατρικής έννοιας. Π.χ. δίνει σαν υπερώνυμο του (vasoconstriction, μείωση της διαμέτρου των αγγείων) το "στένωση" (constriction). Αυτό δίνει έμφαση στο φυσικό μηχανισμό και όχι στην παθολογία, και σαν συνέπεια δεν υπάρχει κάποια συσχέτιση του "vasoconstriction" με τον ιατρικό τομέα στο WordNet.

3.3. Παραδείγματα Ιατρικών οντολογιών

3.3.1. GALEN

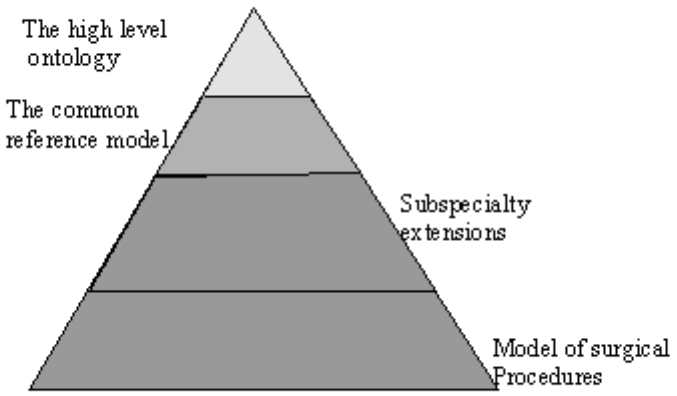
Το GALEN (Generalized Architecture for Languages, Encyclopedias, and Nomenclature in medicine) [23] είναι ένα ευρωπαϊκό έργο που αποσκοπεί στην επαναχρησιμοποίηση πηγών για κλινικά συστήματα. Μια οντολογία, η Common Reference Model, είναι διατυπωμένη σε μια εξειδικευμένη περιγραφική λογική (description logic), την GALEN Representation and Integration Language (GRAIL), και είναι ένα από τα κεντρικά χαρακτηριστικά του GALEN. Η οντολογία αυτή στοχεύει στην αναπαράσταση ιατρικών εννοιών, ανεξάρτητα από κάθε εφαρμογή. Το OpenGALEN παρέχει σημείο πρόσβασης στην οντολογία όπως επίσης και περιγραφές και προδιαγραφές της GALEN τεχνολογίας.

Επιδίωξη του GALEN CRM ήταν να σχεδιαστεί έτσι ώστε να μπορεί να αποτελεί ένα επαναχρησιμοποιήσιμο, και ανεξάρτητο γλώσσας υλοποίησης, μοντέλο ιατρικών εννοιών. Το μοντέλο αυτό θα αποσκοπούσε στην επαναχρησιμοποίηση πληροφορίας με σκοπό την ενοποίηση και διαλειτουργικότητα μεταξύ ιατρικών φακέλων, συστημάτων λήψεις αποφάσεων, και άλλων κλινικών συστημάτων.

Η δομή της οντολογίας που εμπεριέχεται στο GALEN (GALEN Common Reference Model), φαίνεται σχηματικά στο Σχήμα 3.3. Αποτελείται από τα εξής τέσσερα τμήματα:

- Την high level ontology – η οποία περιγράφει τη γενική δομή – τα είδη των concepts, τα οποία αποτελούν μια πιο ευρεία εικόνα του πεδίου, και τα οποία μπορούν να αποσυντεθούν σε πιο λεπτομερειακά concepts.

- Το ίδιο το Common Reference Model – που εμπεριέχει τα επαναχρησιμοποιήσιμα μέρη της ανατομίας, της χειρουργικής, των ασθενειών, των κλινικών ενδείξεων κ.α., τους ορισμούς τους, τις περιγραφές τους και τους περιορισμούς που πρέπει να ισχύουν μεταξύ τους, ώστε να μπορέσουν να συνυπάρξουν μαζί. Το τμήμα αυτό της οντολογίας είναι επίσης όσο πιο ευρύ και ρηχό γίνεται, ώστε να μπορεί να διαμοιράζεται μεταξύ όσο περισσότερων εφαρμογών γίνεται.
- Αναλυτικές επεκτάσεις των δομικών μπλοκ, που είναι απαραίτητες, για συγκεκριμένες εφαρμογές ή συγκεκριμένα υπο-πεδία – που περιέχουν πιο αναλυτικές πληροφορίες για συγκεκριμένες περιοχές του σώματος, τύπους επεμβάσεων κ.α. που χρειάζονται σε συγκεκριμένα υπο-πεδία της χειρουργικής π.χ. στην καρδιοαγγειακή, ουρολογική, αναπνευστική κ.α..
- Το μοντέλο των χειρουργικών μεθόδων, καθώς και άλλα τέτοιου είδους μοντέλα, που ορίζουν σύνθετα concepts που προκύπτουν από την χρήση του Common Reference Model και τις επεκτάσεις του.



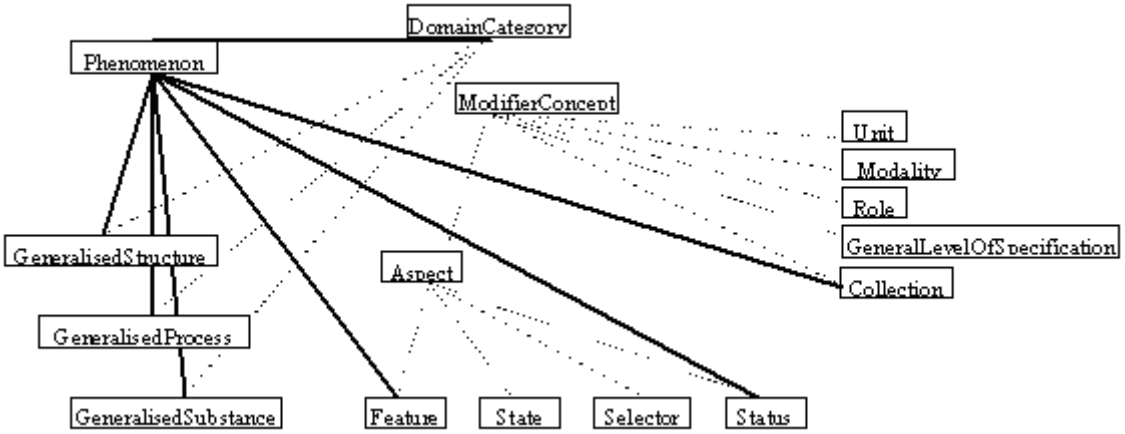
Σχήμα 3.3 Η Δομή της Οντολογίας του GALEN. [71]

Στο Σχήμα 3.4 δίνεται η high level οντολογία του GALEN. Εδώ έχουμε τις εξής βασικές κατηγορίες:

GeneralisedStructures — Αφηρημένες ή φυσικές δομές, που δεν αλλάζουν με το χρόνο
GeneralisedSubstances — Συστατικά, που δεν αλλάζουν με το χρόνο

GeneralisedProcesses — Αλλαγές που μπορούν να προκληθούν με το χρόνο

ModifierConcepts — Ιδιότητες και χαρακτηριστικά των προηγούμενων κατηγοριών



Σχήμα 3.4 Το Υψηλό Επίπεδο της Οντολογίας του GALEN. [71]

Στον Πίνακα 3.1.αναλύεται λίγο περισσότερο η προηγούμενη κατηγοριοποίηση:

Πίνακας 3.1 Ανάλυση των Κατηγοριών της Υψηλού Επιπέδου Οντολογίας του GALEN [71]

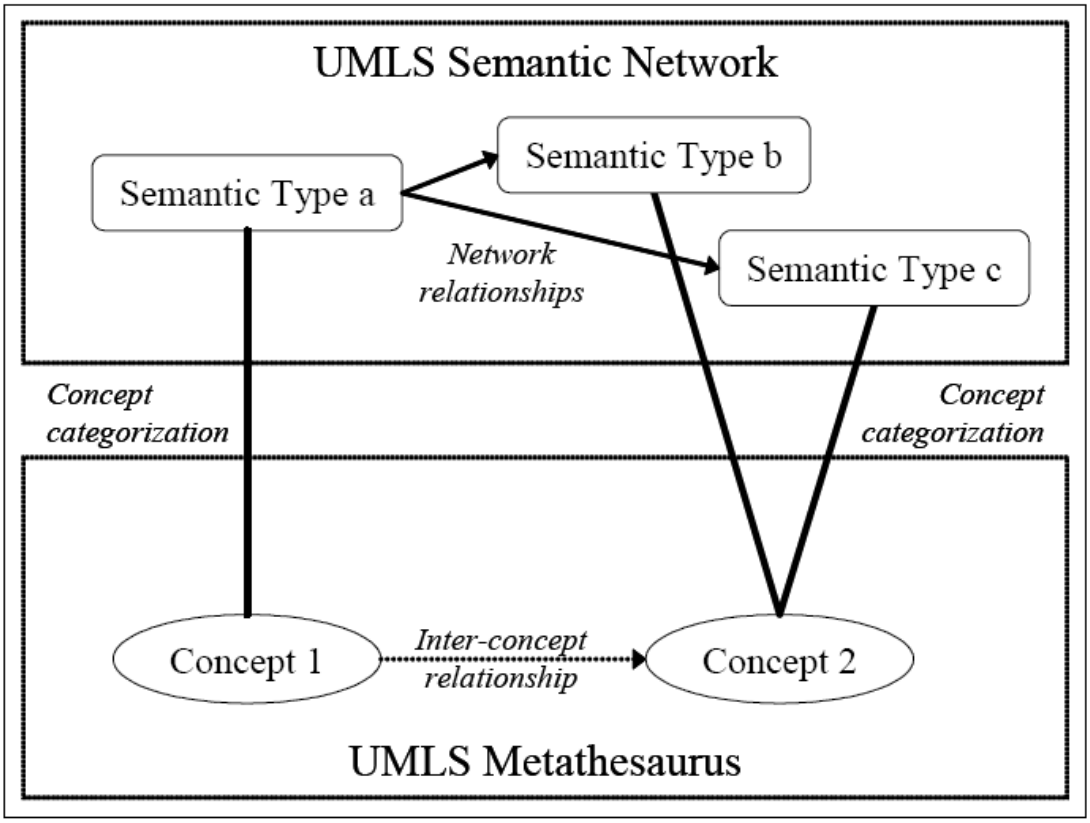
Entity	**Example**
GeneralisedProcess	
SpecificProcess	
BiologicalProcess	
BodyProcess	Peristalsis, Breathing, Clotting
Behaviour	VolitionalAct, ClinicalAct
NonBiologicalProcess	
PhysicalProcess	Irradiation
ChemicalProcess	Histological Staining
GenericProcess	Transport, Opening, Closing
GeneralisedStructure	
AbstractStructure	
PsychosocialConstruct	Clinic, Hospital

LogicalStructure	Protocol, Plan
PhysicalStructure	
LinearStructure	Route
PlanarStructure	Triangle, Square
SolidStructure	
MicroscopicStructure	Cell, Microorganism
InertSolidStructure	Building, Device
OrganicSolidStructure	
BodyStructure	
BodyPart	Heart,Leg, Head, Femur, AdrenalGland, Sacrum, PisiformBone,
	Cusp, Horn, Promontory, Artery, Lump, Bursa, Orifice,
GenericBodyStructure	Rim, Ridge
Organism	Bacterium. Protozoan, Virus, Fungus, Dog, Bird
SolidRegion	PieceOfLiver.
Space	Cavity, Potential Space, PathologicalCavity
GeneralisedSubstance	
Energy	Radiation, SoundEnergy
Substance	
BodySubstance	
Tissue	MuscleTissue, BoneTissue, Mucosa, Endothelium
NAMEDBodySubstance	Urine, Bile , Sputum, Vomit, Sputum .
ChemicalSubstance	Drugs, Sodium .
NAMEDSubstance	Air, Wood
ModifierConcept	
Aspect	
Feature	Sex, Chronicity, Shape, Malignancy, Topology, Colour, Permeability .
State	male/female, acute/chronic, round/square, permeable/impermeable .

Selector	leftSelection/rightSelection, medialSelection/lateralSelection
Status	pathological/pysiological normal/nonNormal
Collection	Polyps (as opposed to polyp)
Modality	FamilyHistory, PreviousHistory, presence/absence
Role	DrugRole, HormoneRole, PatientRole
Unit	second, metre, kilogram
LevelOfSpecification	uniquelySpecified

3.3.2. Unified Medical Language System (UMLS)

Το UMLS [72] αναπτύχθηκε από την National Library of Medicine για να βοηθήσει τους ειδικούς στον τομέα της υγείας και τους ερευνητές να έχουν πρόσβαση σε μια ποικιλία πηγών με ιατρική πληροφορία. Έχει ως στόχο την ανάπτυξη πληροφοριακών συστημάτων τα οποία συμπεριφέρονται σαν να μπορούν να κατανοήσουν τη γλώσσα που χρησιμοποιείται στον τομέα της υγείας και της βιοϊατρικής. Αποτελείται από τρία συστατικά, το Metathesaurus, το Semantic Network και τοSpecialist Lexicon. Το Metathesaurous, που αποτελεί μια μεγάλη πηγή εννοιών (concepts), και το σημασιολογικό δίκτυο (semantic network), που είναι ένα περιορισμένο δίκτυο των 135 σημασιολογικών τύπων, ενοποιούν περισσότερες από 1.000.000 έννοιες μέσα από 100 και πάνω λεξικά και λίστες ορολογιών. Στη διαδικασία κατασκευής του Metathesaurous διατηρείται η δομή του κάθε λεξικού που χρησιμοποιείται, και οι ισοδύναμοι όροι, ομαδοποιούνται σε σημασιολογικά μοναδικές έννοιες. Οι σχέσεις μεταξύ των εννοιών, είτε κληρονομούνται από τα ήδη υπάρχοντα λεξικά, είτε παράγονται κάποιες συγκεκριμένες όπου αυτό χρειάζεται. Αντίθετα το σημασιολογικό δίκτυο, αναπτύχθηκε ανεξάρτητα από τα λεξικά που χρησιμοποιήθηκαν στο metathesaurous, και εξυπηρετεί σαν μια υψηλού επιπέδου οντολογία για τον τομέα της ιατρικής.. Όπως φαίνεται Σχήμα 3.5, οι σημασιολογικοί τύποι του σημασιολογικού δικτύου, χρησιμοποιούνται για την κατηγοριοποίηση όλων των εννοιών του UMLS.



Σχήμα 3.5 Σχηματική Αναπαράσταση του Σημασιολογικού Δικτύου και Metathesaurus του UMLS. [71]

3.3.2.1. Το Metathesaurus

Το Metathesaurus είναι μία πολύ μεγάλη βάση λεξικών. Σήμερα περιλαμβάνει περισσότερα από 1.000.000 concepts και 5.000.000 ονομασίες για αυτά τα concepts, που τα αντλεί από 100 και περισσότερα διαφορετικά λεξικά. Τα λεξικά αυτά ονομάζονται source vocabularies. Κάθε concept έχει κάποια σύνδεση με τα άλλα δύο συστατικά του UMLS, που παρέχει επιπρόσθετες πληροφορίες.

Το Metathesaurus αποδίδει και διατηρεί το νόημα, τις ονομασίες των concepts, και τις σχέσεις από τα διάφορα λεξικά που συμπεριλαμβάνει. Όταν δύο διαφορετικά λεξικά, χρησιμοποιούν το ίδιο όνομα για διαφορετικά concepts, το Metathesaurus αναπαριστά και τα δύο νοήματα και δεικτοδοτεί πιο νόημα βρίσκεται σε πιο λεξικό. Όταν το ίδιο concept εμφανίζεται σε διαφορετικές ιεραρχίες, που οι ιεραρχίες αυτές, αναπαριστούν διαφορετικές σημασιολογίες και βρίσκονται σε διαφορετικά λεξικά, το metathesaurus συμπεριλαμβάνει όλες τις ιεραρχίες. Όταν εμφανίζονται σχέσεις (relationships) μεταξύ δύο concepts, σε

διαφορετικά λεξικά, και αποδίδουν διαφορετική ερμηνεία, τότε και οι δύο οπτικές γωνίες συμπεριλαμβάνονται στο Metathesaurus.

Με άλλα λόγια το Metathesaurus δεν αναπαριστά μια περιεκτική οντολογία, που έχει δημιουργηθεί από την αρχή από την NLM, ή μια απλή οπτική γωνία του πεδίου (εκτός από τους σημασιολογικούς τύπους στο υψηλό της επίπεδο που έχουν δοθεί σε όλα τα concepts). Το Metathesaurus διατηρεί τις πολλές οπτικές γωνίες του πεδίου, που αναπαρίστανται στα διαφορετικά λεξικά που εμπεριέχει, γιατί αυτές οι διαφορετικές οπτικές γωνίες μπορεί να φανούν χρήσιμες σε διαφορετικές εργασίες.

Τα βασικά συστατικά του Metathesaurus, είναι τα concepts, τα strings, τα atoms και οι relations. Κάθε ένα από αυτά έχει και τον δικό του μοναδικό προσδιοριστή (identifier).

Concepts και Concepts Identifiers (CUI)

Στο Metathesaurus τα concepts αναπαριστούν κάποιο νόημα, και ένα νόημα δεν αποδίδεται πάντα από έναν όρο και μόνο. Αυτός είναι και ο λόγος που προσπαθεί να κρατήσει όλα τα συνώνυμα που αποδίδουν το ίδιο νόημα ενωμένα σε concepts.

Κάθε concept έχει έναν μοναδικό προσδιοριστή (concept unique identifier ή CUI). Τα CUI δεν αλλάζουν ποτέ, παρά μόνον όταν δύο ή περισσότερα CUIs αναφέρονται στο ίδιο concept, με άλλα λόγια όταν βρεθεί πως δύο concepts αποτελούν συνώνυμα το ένα ως προς το άλλο, οπότε και το ένα CUI καταργείται.

Concepts names και String Identifiers (SUI)

Τα concepts έχουν πολλές ονομασίες που αποδίδουν το νόημά τους. Κάθε τέτοια ονομασία αποτελεί ένα sting, και κάθε ένα από αυτά έχει και τον μοναδικό προσδιοριστή του (SUI). Για παράδειγμα το ίδιο string σε διαφορετική γλώσσα θα έχει και διαφορετικό SUI.

Atoms και Atoms Identifiers (AUI)

Κάθε string από κάθε λεξικό, αποτελεί και ένα atom για το λεξικό αυτό. Καθένα από αυτά έχει επίσης το μοναδικό προσδιοριστή του (AUI). Για παράδειγμα, αν έχουμε ένα string "Atrial Fibrillation", τότε έχουμε και τα atoms που συνδέονται στο string από τα διαφορετικά λεξικά που το εμπεριέχουν.

Terms και Lexical Identifiers (LUI)

Προς το παρόν οι terms και lexical identifiers είναι διαθέσιμοι μόνο για την αγγλική γλώσσα. Οι λεκτικές ποικιλομορφίες και οι διάφορες διαφοροποιήσεις αποθηκεύονται μαζί σε μία μονάδα. Αυτό σημαίνει πως ένα LUI μπορεί να εμπεριέχει πολλά SUI.

Στο Σχήμα 3.6 φαίνεται η χρησιμότητα των atoms, strings, terms και concepts. Το "Atrial Fibrillation" εμφανίζεται σε περισσότερα από ένα λεξικά, και για κάθε εμφάνισή του θα του δοθεί και ένα AUI. Τα AUI αυτά θα συνδεθούν σε ένα μοναδικό SUI. Ο Πληθυντικός του "Atrial Fibrillation" έχει ένα δικό του διαφορετικό SUI, αλλά αφού τα δύο αυτά αποτελούν λεκτικές ποικιλομορφίες, είναι συνδεδεμένα στον ίδιο term (LUI). Υπάρχει επίσης ένας διαφορετικός όρος (term), ο "Auricular Fibrillation",αλλά αυτός λαμβάνεται σαν συνώνυμος του "Atrial Fibrillations" και έτσι συνδέεται στον ίδιο concept identifier (CUI).

Relationships και Relationship Identifiers

Υπάρχουν πολλές relationships μεταξύ concepts. Αυτές προέρχονται από τα διαφορετικά λεξικά, από τους χρήστες του Metathesaurus, και από τους ανθρώπους της NLM. Υπάρχουν δύο κατηγορίες:

- **Οι Intra-Source:** Αυτές προέρχονται από τα ίδια τα λεξικά, και αναπαριστούν τις σχέσεις των concepts μέσα στο κάθε λεξικό. Για παράδειγμα μπορεί να συνδέουν διαφορετικά concepts, όπως ασθένειες και φάρμακα.
- **Οι Inter-Source:** Αυτές αναπαριστούν τις συσχετίσεις μεταξύ των λεξικών.

Κάθε σχέση έχει τον μοναδικό της προσδιοριστή επίσης (RUI).

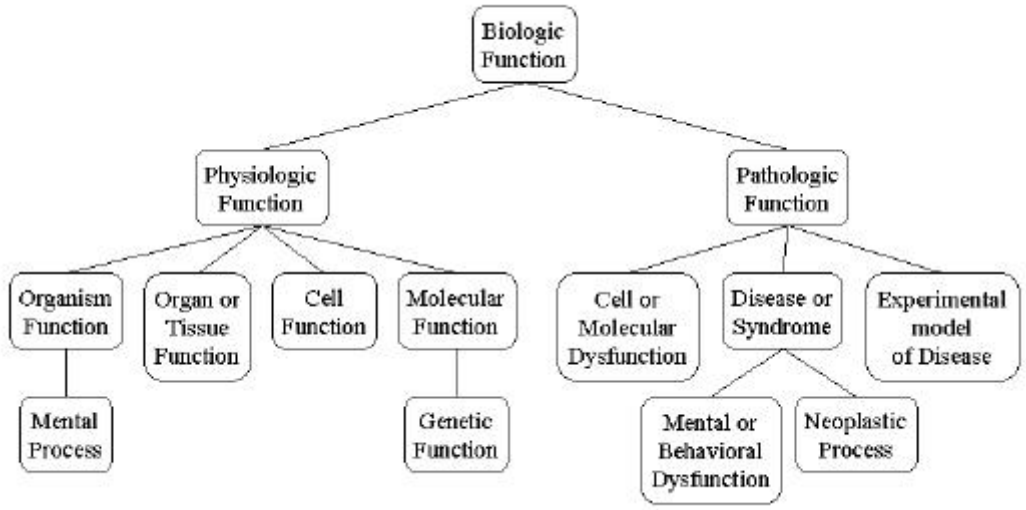
Concept (CUI)	Terms (LUIs)	Strings (SUIs)	Atoms (AUIs) * RRF Only
C0004238 Atrial Fibrillation (preferred) Atrial Fibrillations Auricular Fibrillation Auricular Fibrillations	**L0004238** Atrial Fibrillation (preferred) Atrial Fibrillations	**S0016668** Atrial Fibrillation (preferred)	**A0027665** Atrial Fibrillation (from MSH) **A0027667** Atrial Fibrillation (from PSY)
		S0016669 Atrial Fibrillations	**A0027668** Atrial Fibrillations (from MSH)
	L0004327 (synonym) Auricular Fibrillation Auricular Fibrillations	**S0016899** Auricular Fibrillation (preferred)	**A0027930** Auricular Fibrillation (from PSY)
		S0016900 (plural variant) Auricular Fibrillations	**A0027932** Auricular Fibrillations (from MSH)

Σχήμα 3.6 Παράδειγμα Χρήσης των Concepts, Strings, Atoms και Terms. [72]

3.3.2.2. Semantic Network

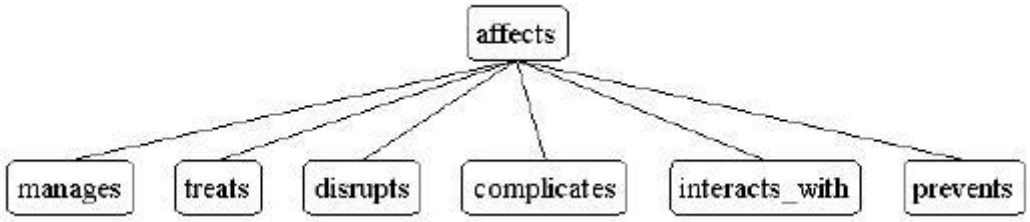
Το σημασιολογικό δίκτυο του UMLS ;έχει ως βασικό σκοπό να παρέχει επιπρόσθετη πληροφορία για τις συσχετίσεις που έχουν μεταξύ τους τα concepts του Metathesaurus, όπως επίσης να τα κατηγοριοποιήσει. Περιέχει 135 σημασιολογικούς τύπους και 54 σχέσεις. Είναι οργανωμένα σε έναν κατευθυνόμενο γράφο, όπου οι σημασιολογικοί τύποι αναπαρίστανται με τους κόμβους και οι σχέσεις με τις ακμές. Μαζί οι σημασιολογικοί τύποι και οι σχέσεις έχουν μια ιεραρχική δομή όπως φαίνεται στο Σχήμα 3.7 και 3.8. Οι σημαντικότεροι τύποι στο σημασιολογικό δίκτυο είναι οι organisms, anatomical structures, biological functions, chemicals, events, physical objects και concepts ή ideas.

Κάθε concept του Metathesaurus τοποθετείται σε τουλάχιστον έναν σημασιολογικό τύπο. Ο ποιο κοντινός τύπος του σημασιολογικού δικτύου δίνεται στο κάθε concept. Για παράδειγμα το “chimpanzee (χιμπατζής)” κατηγοριοποιείται σαν “mammal (θηλαστικό)” και όχι σαν “primate (πρωτεύον θηλαστικό)” αφού δεν υπάρχει σημασιολογικός τύπος για το δεύτερο.



Σχήμα 3.7 Ιεραρχική Δομή του Σημασιολογικού Τύπου: "Biologic Function". [72]

Στο Σχήμα 3.7 βλέπουμε ένα μικρό μέρος του δικτύου και πιο συγκεκριμένα τον σημασιολογικό τύπο "Biologic Function" και τους απογόνους του. Κάθε παιδί συνδέεται με τη σχέση "is-a" με τον πατέρα του.

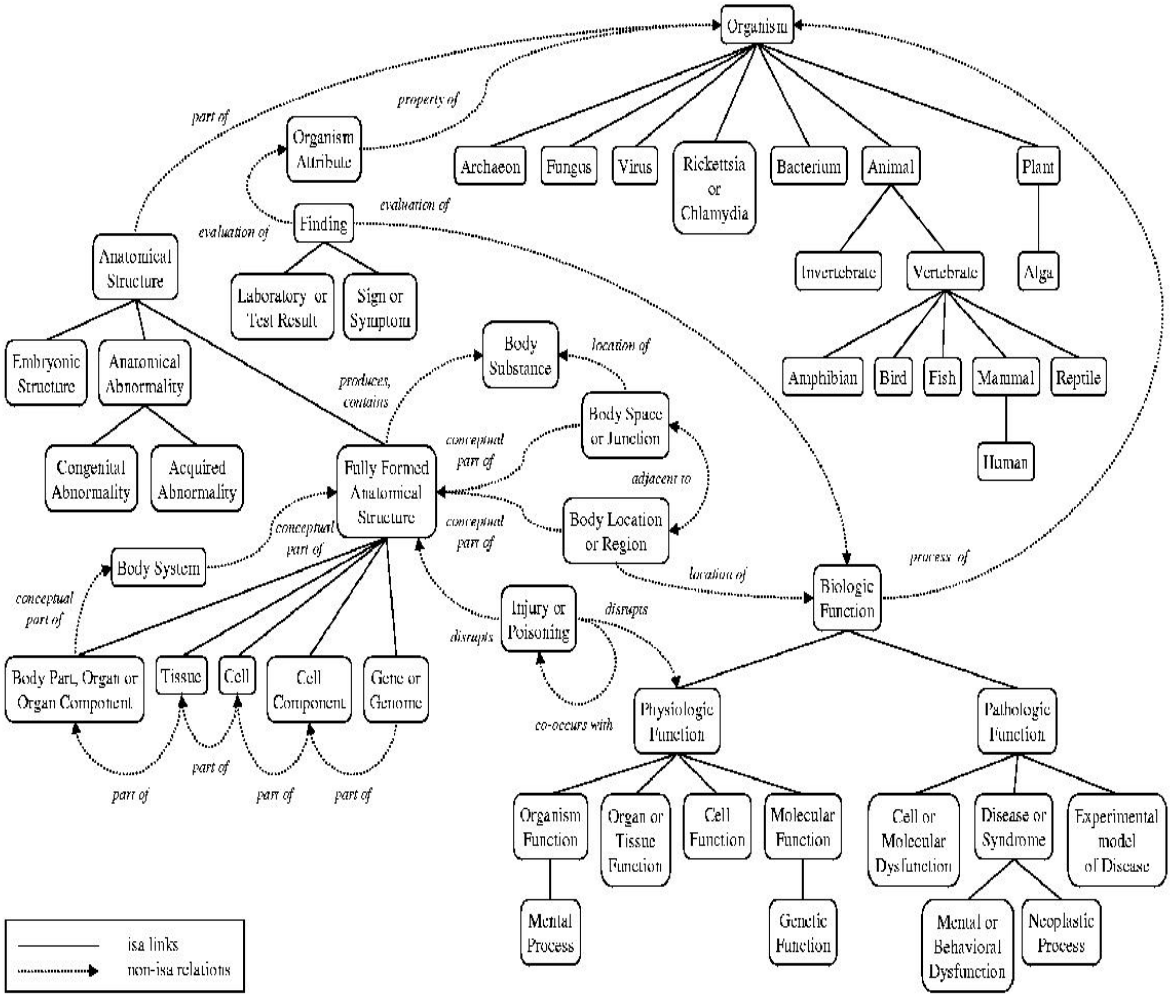


Σχήμα 3.8 Ιεραρχική Δομή της Σχέσης "affects". [72]

Στο Σχήμα 3.8 βλέπουμε την σχέση "affects" και τα παιδιά της.

Όπως φαίνεται στο Σχήμα 3.9, κάθε σημασιολογικός τύπος του υψηλού επιπέδου είναι συνδεδεμένος με μια relationship. Έτσι οι συσχετίσεις κληρονομούνται και στα concepts και instances που ανήκουν στον κάθε σημασιολογικό τύπο. Επίσης υπάρχουν περιπτώσεις όπου μια σχέση κληρονομείται σε χαμηλότερου επιπέδου σημασιολογικούς τύπους αλλά δε στέκει "λογικά" και για το λόγο αυτό μπλοκάρεται. Για παράδειγμα ο τύπος "mental process" δεν

μπορεί να συνδεθεί με τον τύπο"plant" μέσω της σχέσης "process of" αφού τα φυτά δεν είναι σκεπτόμενα όντα.



Σχήμα 3.9 Μέρος του Σημασιολογικού Δικτύου. [72]

3.3.2.3. SPECIALIST Lexicon

Το Lexicon περιέχει λεξικά βιοϊατρικής, καθώς και της κοινής αγγλικής γλώσσας. Για κάθε όρο στο Lexicon η συντακτική, μορφολογική και ορθογραφική πληροφορία καταγράφεται. Η πληροφορία αυτή είναι απαραίτητη για το SPECIALIST Natural Language Processing (NLP) σύστημα. Τα lexical tools χρησιμοποιούν το SPECIALIST NLP σύστημα για να κανονικοποιήσουν τα stings, να ευρετηριοποιήσουν λέξεις και να εντοπίσουν λεκτικές ποικιλομορφίες.

Οι εγγραφές στο Lexicon αποκτούνται από διαφορετικές πηγές, όπως το Webster's Medical Dictionary, το Dorland's Illustrated Medical Dictionary, το Oxford Advanced Learner's Dictionary κ.α.

3.3.3. The Systematized Nomenclature of Medicine (SNOMED)

Η Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT), αναπτύχθηκε από το College of American Pathologists. Είναι μια από τις πιο εκτενείς βιοϊατρικές οντολογίες, που έχουν αναπτυχθεί πρόσφατα σε περιγραφική λογική. Περιέχει περίπου 269.900 κλάσεις, που ονομάζονται από περίπου 407.500 ονόματα. Η SNOMED CT είναι τώρα διαθέσιμη σαν μέρος του UMLS και χρησιμοποιείται ευρέως στα συστήματα ιατρικής πληροφορίας.

Κάθε έννοια της SNOMED CT περιγράφεται από ένα πλήθος στοιχείων. Για παράδειγμα η έννοια "Viral meningitis" έχει έναν μοναδικό προσδιοριστή (58170007), δύο γονικές έννοιες (Infective meningitis και Viral infections of the central nervous system), και πολλές ονομασίες (Viral meningitis, Abacterial meningitis και Aseptic meningitis, viral κ.α.). Οι ρόλοι (ή σημασιολογικές σχέσεις) που παρουσιάζονται στον ορισμό αυτής της έννοιας φαίνονται στον παρακάτω πίνακα:

Πίνακας 3.2 Οι Ρόλοι (Σχέσεις) που δίνει η SNOMED CT στην Έννοια Viral meningitis. [71]

Role	Value
Causative agent	Virus
Associated morphology	Inflammation
Finding site	Meninges structure
Onset	Sudden onset Gradual onset
Severitiy	Severities
Episodicity	Episodicities
Course	Courses

Η SNOMED CT αποτελείται από δεκαοχτώ ιεραρχίες, που αντανακλούν, εν μέρει, την οργάνωση προηγούμενης έκδοσης του SNOMED σε "Άξονες" όπως Diseases, Drugs, Living,

organisms, Procedures και Topography. Οι έννοιες του πρώτου επιπέδου κάθε άξονα δίνονται στο Σχήμα 3.10.

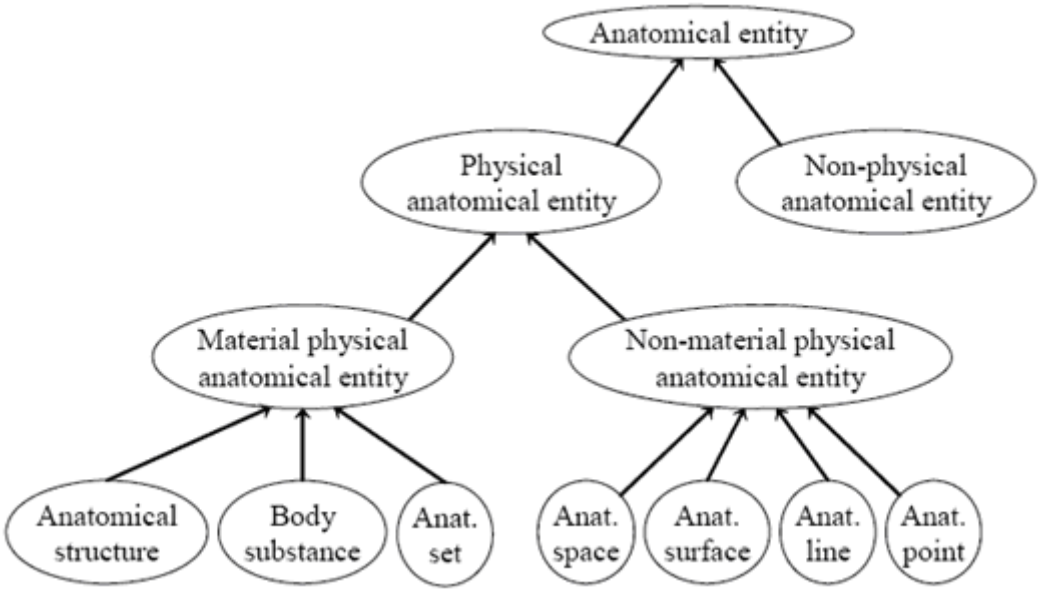
Top-level concepts
Attribute
Body structure
Clinical finding
Context-dependent categories
Environments and geographical locations
Events
Observable entity
Organism
Pharmaceutical / biologic product
Physical force
Physical object
Procedure
Qualifier value
Social context
Special concept
Specimen
Staging and scales
Substance

Σχήμα 3.10 Οι Έννοιες του Πρώτου Επιπέδου των 18 Ιεραρχιών του SNOMED CT. [71]

3.3.4. Foundational Model of Anatomy (FMA)

Η Foundation Model of Anatomy, αναπτύχθηκε από το πανεπιστήμιο της Washington, με σκοπό να ενισχύσει το σχετικό με ανατομία περιεχόμενο του UMLS. Επικεντρώνοντας αποκλειστικά στην αναπαράσταση της δομής, η FMA αναμένεται να εξυπηρετήσει σαν αναφορική οντολογία, π.χ. να επιτρέψει σε άλλες οντολογίες που έχουν την ανατομία σαν συστατικό τους, να ευθυγραμμιστούν μαζί της. Συγκεκριμένα, στόχος της FMA είναι να παρέχει μια περιγραφή των υλικών αντικειμένων, και χωρικών πληροφοριών που αποτελούν το ανθρώπινο σώμα. Ενοποιεί την Anatomical Ontology με δύο μικρότερες οντολογίες, την Physical State Ontology και την Spatial Ontology. Η δεύτερη αναπαριστά γεωμετρικά αντικείμενα και κλάσεις τρισδιάστατων σχημάτων. Επίσης η Spatial Ontology, κάνει διαφοροποίηση μεταξύ πραγματικών και εικονικών ορίων των όγκων, των επιφανειών και των γραμμών. Η Anatomical Ontology, περιέχει περίπου 70.000 concepts, που αρχικά περιορίζονταν στην χοντρική ανατομία (gross anatomy), και πλέον αρχίζει να επεκτείνεται σε κυτταρικά και υπο-κυτταρικά φαινόμενα. Η FMA είναι υλοποιημένη στο Protégé.

Οι ορισμοί των ανατομικών οντοτήτων στην Foundational Model of Anatomy, διατυπώνονται ορίζοντας περιορισμούς, βασιζόμενους σε χωρικές διαστάσεις, στη μάζα, σε τρισδιάστατα σχήματα, όπως επίσης σε δομικές μονάδες που αποτελούν το σώμα. Οι relationships ωστόσο, περιορίζονται στη δομική οργάνωση των φυσικών ανατομικών οντοτήτων. Το πιο υψηλό επίπεδο της οντολογίας είναι η Anatomical entity, η οποία χωρίζεται σε Physical anatomical entity και Non-physical anatomical entities (Σχήμα 3.11).



Σχήμα 3.11 Το Υψηλό Επίπεδο της FMA. [71]

Οι Physical entities έχουν χωρικές διαστάσεις, ενώ οι non-physical, όπως το στάδιο εξέλιξης (developmental stage), δεν έχουν. Επιπλέον διαχωρισμός γίνεται μεταξύ των physical entities που έχουν μάζα, όπως οι ανατομικές δομές και τα συστατικά του σώματος (Material physical anatomical entity), και αυτών που δεν έχουν, στις οποίες συμπεριλαμβάνονται οι ανατομικοί χώροι, επιφάνειες, γραμμές και σημεία (Non-material physical anatomical entity). Ιεραρχίες στην FMA σχηματίζονται χρησιμοποιώντας την μεταβατική σχέση “part-of”, όπως επίσης με τις σχέσεις “branch-of” και “tributary-of”, που αναπαριστούν σχέσεις μεταξύ δομών που μοιάζουν με δέντρο (tree-like) όπως: νεύρα, αρτηρίες, φλέβες και αγγεία.

ΚΕΦΑΛΑΙΟ 4. ΧΡΗΣΗ ΟΝΤΟΛΟΓΙΩΝ ΣΕ ΣΥΣΤΗΜΑΤΑ ΙΑΤΡΙΚΗΣ

Οι οντολογίες σήμερα χρησιμοποιούνται όλο και περισσότερο σε συστήματα ιατρικής. Λόγω του τεράστιου όγκου πληροφορίας και γνώσης που υπάρχουν σε τέτοιου είδους συστήματα γίνεται ολοένα και πιο επιτακτική η ανάγκη για το διαμοιρασμό τους. Επίσης, αναγκαίο γίνεται όλο και περισσότερο να υπάρξουν συστήματα που να μπορούν να επικοινωνούν μεταξύ τους χωρίς δυσκολίες, καταλαβαίνοντας το ένα την πληροφορία που του παρέχει το άλλο, δηλαδή ανάγκη για διαλειτουργικότητα.

Οι οντολογίες σε ιατρικά συστήματα παρατηρείται να χρησιμοποιούνται για τη διαλειτουργικότητα συστημάτων, την αναζήτηση σε βάσεις δεδομένων, την εξόρυξη γνώσης από κείμενα καθώς και την ανάκτηση εγγράφων.

Στο κεφάλαιο αυτό θα παρουσιάσουμε μερικές εργασίες που έχουν γίνει στους τομείς αυτούς και που έχουν να κάνουν με τη χρήση οντολογιών στο ιατρικό πεδίο.

4.1. Χρήση Οντολογιών για Σημασιολογική Διαλειτουργικότητα Συστημάτων

Μιλώντας για σημασιολογική διαλειτουργικότητα, εννοούμε το να μπορούν συστήματα που χρησιμοποιούν πληροφορία, που είναι εκφρασμένη με διαφορετικό τρόπο, να μπορούν να επικοινωνήσουν μεταξύ τους, καθώς και με τους ίδιους τους χρήστες τους.

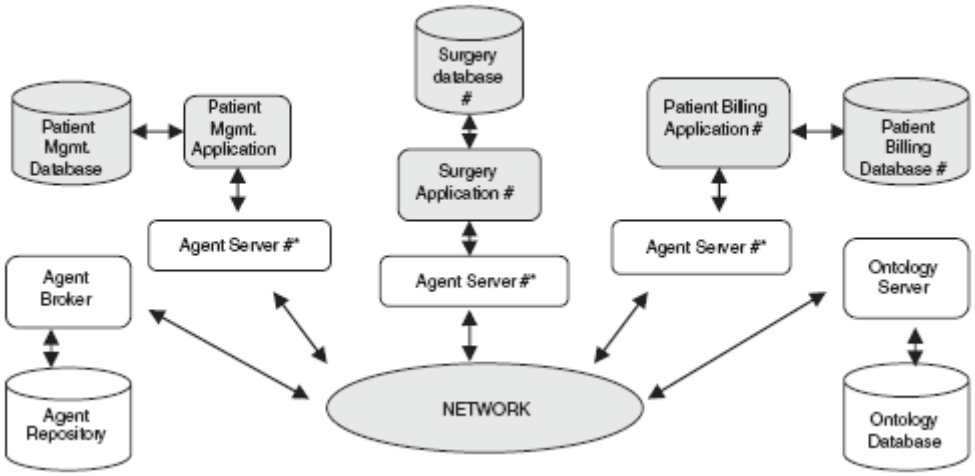
Στη συνέχεια παρατίθενται μια σειρά από εργασίες που έχουν ως στόχο τη σημασιολογική διαλειτουργικότητα, μεταξύ ετερογενών συστημάτων και των χρηστών τους.

Στη δημοσίευση των Claudio Eccher et al. "Ontologies supporting continuity of care: The case of heart failure" [24] παρουσιάζεται μια εφαρμογή που υποστηρίχθηκε από την επαρχία του Trento (North Italy), που στοχεύει στην υλοποίηση ενός web-based συστήματος διαχείρισης φακέλων ασθενών, λόγω της μεγάλης αύξησης των περιστατικών με καρδιακά επεισόδια. Η εφαρμογή αυτή αναπτύχθηκε βασιζόμενη στην openEHR αρχιτεκτονική όπου γίνεται χρήση των archetypes [25] που προτάθηκε από την openEHR foundation [26]. Τα archetypes είναι ένα επαναχρησιμοποιήσιμο, formal μοντέλο μιας έννοιας (concept) του πεδίου ενδιαφέροντος. Ορίζουν το περιεχόμενο και τη δομή που πρέπει να έχουν τα δεδομένα που αντιστοιχούν σε ένα concept, σε τεχνικούς όρους ένα archetype ορίζει περιορισμούς στη δομή των δεδομένων. Τα archetypes διαφοροποιούνται από τις οντολογίες στο ότι μοντελοποιούν πληροφορία όπως αυτή χρησιμοποιείται από συγκεκριμένους χρήστες και για συγκεκριμένο σκοπό (πληροφορία που έχει να κάνει με τον ηλεκτρονικό φάκελο ασθενούς), ενώ οι οντολογίες έχουν ως σκοπό την αναπαράσταση της πραγματικότητας. Έτσι, ως παράδειγμα το archetype για την "systematic arterial blood pressure measurement" είναι ένα μοντέλο για την πληροφορία που υπάρχει σε αυτού του είδους τη μέτρηση: συνήθως systolic και diastolic pressure, την κατάσταση του ασθενή (συνθήκες μέτρησης, επίπεδο άσκησης) και όργανα που χρησιμοποιήθηκαν. Από την άλλη μια οντολογία θα επικεντρωθεί στο να περιγράψει με λιγότερη ή περισσότερη λεπτομέρεια τι είναι η blood pressure (σαν ένα φυσικό φαινόμενο). Η οντολογία για την "blood pressure" θα μας δώσει πληροφορία για το φυσικό φαινόμενο, τον υπολογισμό του, το αποτέλεσμα αυτού του υπολογισμού και τους συμμετέχοντες σε αυτή την διαδικασία (πχ. γιατρούς, όργανα κ.α). Ακόμη και χωρίς την μέτρηση, χωρίς τον φάκελο ασθενούς και χωρίς τους συμμετέχοντες και τα όργανα που παίρνουν μέρος στην αξιολόγησή της, για μια οντολογία το φυσικό φαινόμενο θα συνεχίσει να υπάρχει. Τα archetypes έχουν διαφορετικό σκοπό και δεν αναδεικνύουν τόση σημασιολογική ακρίβεια. Οι συγγραφείς δίνουν μεγάλη έμφαση στο ρόλο των οντολογιών

στην υποστήριξη της «συνεχής φροντίδας». Σε ένα περίπλοκο σενάριο όπου πολλοί εμπλεκόμενοι (agents) συνεργάζονται για να επιτραπεί η «συνεχής φροντίδα», οι οντολογίες είναι το ουσιαστικό μέσο που σιγουρεύει τη σημασιολογική συνέπεια των δεδομένων και της γνώσης, που διαμοιράζονται μεταξύ διάφορων προσώπων που εμπλέκονται στις διάφορες διαδικασίες, μεταξύ των οποίων και οι ασθενείς καθώς και οι οικογένειές τους, και βοηθούν στην αποφυγή της μη σωστής κατανόησης της σημασιολογίας της πληροφορίας που χρειάζεται να διακινηθεί μεταξύ αυτών. Έτσι, προτείνουν την εξέλιξη του φακέλου ασθενούς από την κλασική openEHR αρχιτεκτονική που είναι βασισμένη σε archetypes σε μια που να έχει ως βάση μια οντολογία. Κάθε όρος του φακέλου ασθενούς θα αντιστοιχίζεται σε έννοιες μιας οντολογίας του πεδίου της ιατρικής και έτσι θα μπορεί να αποφευχθεί κάθε είδους σύγχυση που μπορεί να προκύψει στη σημασιολογία του. Κάτι τέτοιο βοηθά στο διαμοιρασμό της γνώσης, δηλαδή στη δημιουργία διαλειτουργικών συστημάτων ηλεκτρονικών φακέλων ασθενών.

Οι B. Orgun και J. Vu στην εργασία τους «HL7 ontology and mobile agents for interoperability in heterogeneous medical information systems» [27], αναφέρουν τη μεγάλη ανάγκη υψηλής διαλειτουργικότητας μεταξύ διάφορων οντοτήτων διαχείρισης ιατρικής πληροφορίας και γνώσης. Αναφέρουν πως τα συστήματα πολλαπλών agents, βασισμένα σε οντολογίες, παρέχουν ένα καλό πλαίσιο για αλληλεπίδραση σε ένα κατανεμημένο περιβάλλον ιατρικών συστημάτων χωρίς τους περιορισμούς που βάζουν οι πιο παραδοσιακές τεχνικές της μορφής client-server. Στην εργασία αυτή παρουσιάζεται ένα σύστημα πολλαπλών agents με μια οντολογία, το electronic Medical Agent System (eMAGS), βασισμένο στο αποδεκτό στάνταρτ, Health Level Seven (HL7), που διευκολύνει τη ροή της πληροφορίας ασθενών μεταξύ ενός οργανισμού υγείας. Η οντολογία αυτή είναι βασισμένη στο HL7-RIM (HL7 Reference Information Model), που περιγράφει τις έννοιες που συμπεριλαμβάνονται στα HL7 μηνύματα και τις σχέσεις μεταξύ τους. Η αρχιτεκτονική του eMAGS (Σχήμα 4.1) αποτελείται από ένα σύνολο πρακτόρων εξυπηρετητών (agent servers), που ο καθένας εξυπηρετεί μια εφαρμογή βάσης δεδομένων, από αυτές που αποτελούν μέρος του eMAGS δικτύου, από έναν πράκτορα μεσάζον (agent broker) και έναν εξυπηρετητή οντολογίας (ontology server). Έτσι με τη βοήθεια της οντολογίας που έχουν υλοποιήσει, η οποία υλοποιήθηκε στο protégé και διατηρείται σε RDF/RDFS μορφή, μπορεί να γίνεται μια αναζήτηση κοινών εννοιών ή εννοιών που συνδέονται μεταξύ τους με κάποια σχέση, μέσω της σημασιολογίας του HL7 και να πραγματοποιείται μια αντιστοίχηση (mapping) μεταξύ

των όρων που χρησιμοποιούνται στα πρότυπα των μηνυμάτων του HL7 (HL7 message template library HL7MLib) και των όρων που υπάρχουν στις διαφορετικές εφαρμογές βάσεων δεδομένων, που διατηρούνται από διαφορετικούς οργανισμούς και έχουν ενσωματωθεί στο σύστημα, και οι οποίοι διαφέρουν στον τρόπο αποθήκευσης της πληροφορίας. Έτσι επιτυγχάνεται η σωστή μετατροπή μέσω των agents, της πληροφορίας που κατέχει ο κάθε οργανισμός, στη μορφή που απαιτεί το HL7 για επιτυχημένη διάδοσή της στους υπόλοιπους οργανισμούς που μετέχουν στο σύστημα.

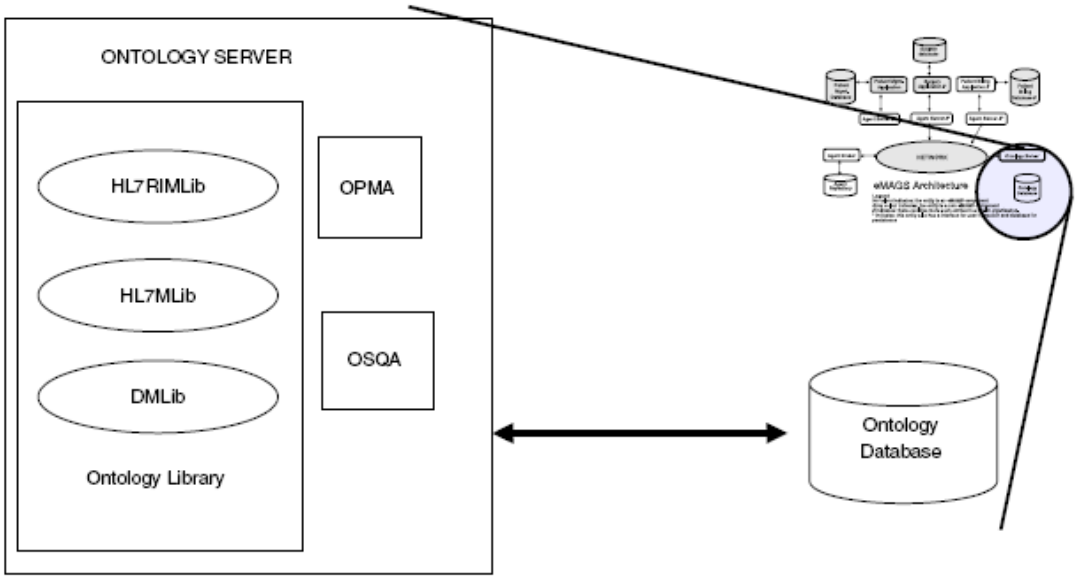


Σχήμα 4.1 Η Αρχιτεκτονική του eMAGS. [27]

Στο Σχήμα 4.2 βλέπουμε τα επιμέρους συστατικά του ontology server τα οποία είναι τα εξής:

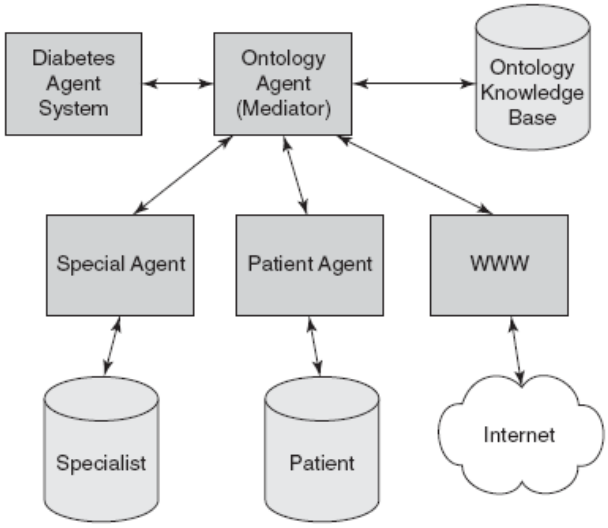
- HL7 message template library (HL7MLib) που περιέχει τα HL7 message formats για τα ποικίλα HL7 trigger events.
- Την οντολογία που είναι βασισμένη στο HL7-RIM (HL7RIMLibrary ή HL7RIMLib).
- Μια βιβλιοθήκη αντιστοιχίσεων (mapping library, DMLib) που περιέχει μια λίστα με αντιστοιχίσεις μεταξύ των πεδίων διάφορων βάσεων δεδομένων και των πεδίων των HL7 messages.
- Έναν query agent για την ανάκτηση πληροφοριών από τις ποικίλες ontology libraries (Ontology Server Query Agent—OSQA).
- Μια διαπροσωπεία χρήστη (user interface agent) που επιτρέπει σε κάποιο διαχειριστή να έρχεται σε επαφή με τα δεδομένα του ontology server και έναν process manager

agent για τη διαχείριση όλων των ενεργειών (Ontology Process Manager Agent—OPMA).



Σχήμα 4.2 Τα Επιμέρους Συστατικά του Ontology Server του eMAGS. [27]

Στην εργασία των Probnab Ganguly et al. [28] παρουσιάζουν ένα σύστημα που παρέχει σημασιολογική διαλειτουργικότητα στην τηλεϊατρική. Η τηλεϊατρική εμπλέκει πληροφορίες από ανθρώπους και μηχανήματα, καθώς και διάφορες τεχνολογίες ιατρικής φροντίδας. Σε ένα τέτοιο περιβάλλον που χρειάζεται εφαρμογές να τρέχουν σε ετερογενή περιβάλλοντα, η διαλειτουργικότητα του λογισμικού είναι επιτακτική. Οι συγγραφείς εδώ προτείνουν μια επίλυση στη σημασιολογική διαλειτουργικότητα, βασισμένη σε οντολογίες και πράκτορες, στο πεδίο της διαχείρισης του διαβήτη. Στην παρακολούθηση και θεραπεία του διαβήτη οι ασθενείς, οι άνθρωποι των ιατρικών εργαστηρίων, οι γιατροί και οι ειδικοί αλληλεπιδρούν μεταξύ τους για να παρασχεθεί αποτελεσματική ιατρική φροντίδα στους ασθενείς. Μια νέα προσέγγιση είναι απαραίτητη να καθιερωθεί για να υπάρξει συνέπεια στα διάφορα ετερογενή περιβάλλοντα. Η αρχιτεκτονική του προτεινόμενου συστήματος φαίνεται στο Σχήμα 4.3.



Σχήμα 4.3 Η Αρχιτεκτονική του Συστήματος Διαχείρισης του Διαβήτη. [28]

Το μοντέλο είναι βασισμένο πάνω σε ένα σύστημα πρακτόρων, που αλληλεπιδρούν μεταξύ τους μέσω ενός πρωτοκόλλου επικοινωνίας πρακτόρων, και μιας οντολογίας. Τα διάφορα συστατικά αναλύονται παρακάτω:

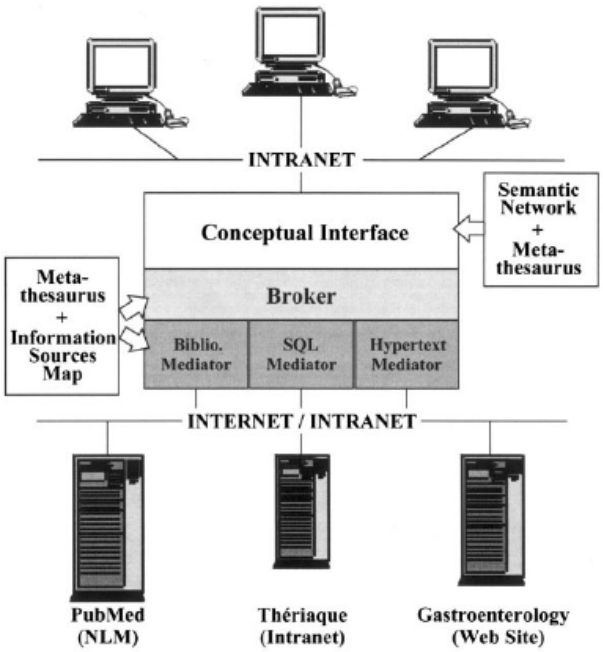
- Diabetes agent system: Παρέχει μια διαπροσωπεία για τον γιατρό ή τον ειδικό. Στέλνει αιτήματα από τον χρήστη στον ontology agent, και λαμβάνει και ανταποκρίνεται σε μηνύματα από τον Ontology agent.
- Ontology agent: Λαμβάνει μηνύματα από κάθε agent του συστήματος. Τα μηνύματα σπάνε σε λεκτικές μονάδες για να μεταφραστούν σε βασικά concepts. Αυτή η λειτουργία γίνεται σε συνεργασία με τη βάση γνώσης. Π.χ. αν ένα μήνυμα περιέχει τη λεκτική μονάδα “web” και το βασικό concept που ορίζεται στην οντολογία είναι “internet”, όλα τα μηνύματα που περιέχουν τη λέξη web θα τη μετατρέψουν σε internet. Ο Ontology agent διερωτά όλους τους agent για το ποιος μπορεί να ικανοποιήσει το μεταφρασμένο μήνυμα και στη συνέχεια στέλνει το μήνυμα στον κατάλληλο agent.
- Specialist agent: Παρέχει πληροφορίες σχετικές με την επιλογή θεραπείας για έναν ασθενή. Παρέχει πρόσβαση σε μια βάση γνώσης για να καθορίσει σωστή επιλογή θεραπείας βασισμένη στα συμπτώματα.

- Patient agent: Ανταποκρίνεται σε αιτήματα για πληροφορίες σχετικές με τον ασθενή, όπως τα επίπεδα γλυκόζης στο αίμα του, συμπτώματα και γενικά δεδομένα που τον αφορούν. Θα πρέπει να αλληλεπιδρά με τον ασθενή για να συγκεντρώνει πληροφορίες.
- World Wide Web agent: Παρέχει πρόσβαση στον παγκόσμιο ιστό και κάνει μια αναζήτηση σε δεδομένα που έχουν ζητηθεί από άλλους agents.

Ένα σενάριο που μας δίνουν οι συγγραφείς για το πώς λειτουργεί το σύστημα είναι το εξής: Ο χρήστης (γιατρός) ζητά μια πληροφορία από τον Diabetes agent. Το αίτημα αυτό στέλνεται στον Ontology agent που το μεταφράζει βασιζόμενος στα βασικά concepts που περιέχονται στην οντολογία. Ο Ontology agent θα διασχίσει κάθε λέξη στο μήνυμα και θα τη μεταφράσει σε ένα βασικό concept. Αυτό διασφαλίζει πως όλα τα μηνύματα που περνάνε από το σύστημα είναι συνεπή και μπορούν να είναι κατανοητά από όλους τους agents. Μετά τη μετάφραση του μηνύματος ο ontology agent πρέπει να προσδιορίσει έναν agent που μπορεί να καταλάβει και να ικανοποιήσει το αίτημα. Για να γίνει αυτό ο ontology agent θα ρωτήσει κάθε agent για να δει ποιος ικανοποιεί τους βασικούς όρους του μηνύματος. Αν βρεθεί κάποιος τότε το μήνυμα προωθείται σε αυτόν. Για παράδειγμα το μήνυμα "find information on the web about diabetes" θα σταλθεί στο www agent, ο οποίος θα ψάξει σε έναν browser για να βρει πληροφορίες για τον διαβήτη.

Οι Michel Joubert et al. [29] μας παρουσιάζουν ένα σύστημα, το οποίο δίνει τη δυνατότητα στους χρήστες να έχουν πρόσβαση σε διάφορες πηγές, που η σημασιολογία των δεδομένων τους είναι ανομοιογενείς μεταξύ τους. Το σύστημα αυτό το ονομάζουν ARIANE και σκοπό έχει να γίνει μια προσπάθεια για έναν ομοιογενή και λειτουργικό τρόπο για το χτίσιμο ενός συστήματος ερωτήσεων (query system), ικανό να ενσωματώνεται σε υπάρχοντα πληροφοριακά συστήματα, και να ενσωματώνει το ίδιο, υπάρχοντες καθώς και νέες πηγές πληροφορίας. Οι σημασιολογικές προσεγγίσεις κάνουν χρήση οντολογιών που αποτελούν μια αναπαράσταση των εννοιών που χρησιμοποιούνται στο εκάστοτε πεδίο, με σχέσεις μεταξύ τους καθώς και κανόνες που περιορίζουν τη σημασία των όρων και επιφέρουν μια συνεπή ερμηνεία τους. Στη συγκεκριμένη εργασία οι συγγραφείς κάνουν χρήση των πηγών γνώσεων που περιλαμβάνονται στο UMLS για να πετύχουν το σκοπό τους. Επίσης χρησιμοποιούν και ένα άλλο συστατικό του UMLS, το Information Sources Map (ISM), με την επιδίωξη να καθοδηγήσουν τους χρήστες στους σωστούς (σημασιολογικά) εξυπηρετητές

(servers). Οι πηγές πληροφορίας που χρησιμοποιούν στο πείραμά τους, είναι α) το PubMed που εξυπηρετεί το Medline, β) η Theriaque, μια βάση δεδομένων που περιλαμβάνει όλα τα φάρμακα που είναι διαθέσιμα στα Γαλλικά νοσοκομεία και γ) μια ιστοσελίδα που βρίσκεται στην ιατρική σχολή του Nice στη Γαλλία, που καταγράφει και δίνει πρόσβαση σε πληροφορίες που έχουν να κάνουν με το πεδίο της γαστρεντερολογίας και της διατροφής. Η πρόσβαση σε καθέναν από αυτούς τους εξυπηρετητές είναι διαφορετική, και εξαρτάται στο είδος της πληροφορίας που διανέμουν και στην τεχνολογία που υποστηρίζουν για να ερωτηθούν. Η αρχιτεκτονική του συστήματος αυτού φαίνεται στην Σχήμα 4.4.



Σχήμα 4.4 Η Αρχιτεκτονική του ARIANE. [29]

Οι συγγραφείς επιδιώκουν τη δημιουργία μιας μεθόδου που θα βοηθά τους τελικούς χρήστες να μην αποπροσανατολίζονται από ανοργάνωτα δεδομένα και δομές πληροφορίας. Ένα από τα μέσα που χρησιμοποιούν είναι η ευρετηριοποίηση της πληροφορίας σύμφωνα με ένα κοινό λεξικό, όπως το MeSH. Αυτό είναι εφικτό στις μέρες μας αφού το MeSH αποτελεί έναν παγκόσμια αναγνωρισμένο θησαυρό, που είναι διαθέσιμος για βιβλιογραφικούς εξυπηρετητές και σχεσιακές βάσεις, καθώς επίσης η κοινότητα του διαδικτύου προτείνει τη χρήση μετα-δεδομένων στις κεφαλίδες των HTML σελίδων. Η διαπροσωπεία μεταξύ του τελικού χρήστη και των πηγών πληροφορίας θα είναι οργανωμένη σύμφωνα με τη σημασιολογία που παρέχεται από μια οντολογία. Αφού το UMLS Semantic Network (SN) παρέχει ένα επαρκές

σύνολο σχέσεων μεταξύ των εννοιών, η μέθοδος που προτείνουν οι συγγραφείς είναι σύμφωνη με αυτή την αναπαράσταση γνώσης. Επιπλέον το Metathesaurous του UMLS παρέχει τα πλέον μεγαλύτερα και πιο ενημερωμένα λεξικά στην βιοιατρική. Έτσι το SN και το Metathesaurous μπορούν να ληφθούν σαν βάση για την υλοποίηση μιας λειτουργικής, εννοιολογικής διαπροσωπείας μεταξύ του τελικού χρήστη και του Broker. Έτσι αν το ενδιαφέρον ενός χρήστη είναι η θεραπεία του γαστρικού έλκους με ρανιτιδίνη ο εννοιολογικός γράφος που θα σχηματιστεί είναι ο εξής: [Pharmacological substance: ranitidine] -> (treats) ->[Disease or syndrome: gastric ulcer], όπου το "Pharmacological substance" και το "Disease or syndrome" είναι οι σημασιολογικοί τύποι του SN στους οποίους συνδέονται τα "ranitidine" και "gastric ulcer" αντίστοιχα στο UMLS.

Η μέθοδος που προτείνουν είναι η εξής:

- ο χρήστης δίνει τις λέξεις για τις οποίες θέλει να βρει ένα concept στο UMLS.
- η εννοιολογική διαπροσωπεία επιστρέφει μια λίστα με concepts που περιέχουν αυτές τις λέξεις. Ο χρήστης επιλέγει το concept που τον ενδιαφέρει.
- Η εννοιολογική διαπροσωπεία επιστρέφει μια λίστα με τα συμφραζόμενα στα οποία το concept αυτό μπορεί να ανταποκρίνεται. Ο χρήστης επιλέγει το συμφραζόμενο του concept που επιθυμεί.
- Στα βήματα (4), (5) και (6) γίνεται η αντίστοιχη διαδικασία για την επιλογή μια άλλης έννοιας. Μέχρι και το 6^{o} βήμα γίνεται χρήση πληροφορίας που βρίσκεται στο Metathesaurus μόνο.
- (7) Το σύστημα εντοπίζει τους σημασιολογικούς τύπους στους οποίους αντιστοιχίζεται το κάθε concept και μέσω του Semantic Network εντοπίζει τις σχέσεις που υφίστανται μεταξύ των concepts και τις επιστρέφει στο χρήστη. Μόλις ο χρήστης επιλέξει μια σχέση, έχει σχηματιστεί ένα στοιχειώδες ερώτημα όπως το "gastric ulcer treated by ranitidine".
- (8) ο έλεγχος πλέον πηγαίνει στον Broker. Αυτός αξιοποιεί το ερώτημα για να βρει σχετικές διαθέσιμες πηγές, ώστε να οδηγήσει το ερώτημα του χρήστη σε έναν σημασιολογικά κατάλληλο εξυπηρετητή. Για να το πετύχει αυτό χρειάζεται πληροφορία από δύο πηγές γνώσης: γνώση για να μεταφράσει τη σημασιολογική σχέση μεταξύ των concepts σε MeSH subheadings που συσχετίζονται με αυτά τα

concepts. π.χ. η σχέση “treats” μεταξύ μιας ασθένειας και του φαρμάκου της, όπως στο παράδειγμα του παραπάνω ερωτήματος, θα μεταφραστεί σε μια λίστα από keywords που θα αντιστοιχούν σε MeSH subheadings, έτσι θα έχουμε την εξής λίστα, [ranitidine]/therapeutic use, pharmacokinetics, administration and dosage, pharmacology [gastric ulcer]/drug therapy. Επίσης ο Broker χρειάζεται πληροφορία από το ISM. Στο ISM κάθε εξυπηρετητής περιγράφεται με διάφορα πεδία όπως το όνομά του, τη γλώσσα του, μια περιγραφή του, το πρωτόκολλο με το ποίο μπορεί να γίνει επικοινωνία μαζί του κ.α.. Η περιγραφή του μπορεί να γίνει με MeSH keywords και subheadings που δεικτοδοτούν το είδος της πληροφορίας που διανέμουν. Μια διαδικασία αντιστοίχησης μεταξύ της αναπαράστασης του ερωτήματος και της περιγραφής των εξυπηρετητών επιτρέπει στον Broker να επιστρέψει στον χρήστη μια λίστα με τις κατάλληλες πηγές πληροφορίας που συσχετίζονται με το ερώτημά του.

- (9) ο Broker επιστρέφει στο χρήστη μια λίστα των κατάλληλων εξυπηρετητών που μπορούν να του δώσουν απαντήσεις στο ερώτημά του. Με την επιλογή ενός εξυπηρετητή ενεργοποιείται ο κατάλληλος mediator ο οποίος δημιουργεί ένα κατάλληλο ερώτημα για να επικοινωνήσει με τον συγκεκριμένο εξυπηρετητή, και επιστρέφει τα αποτελέσματα στον χρήστη.

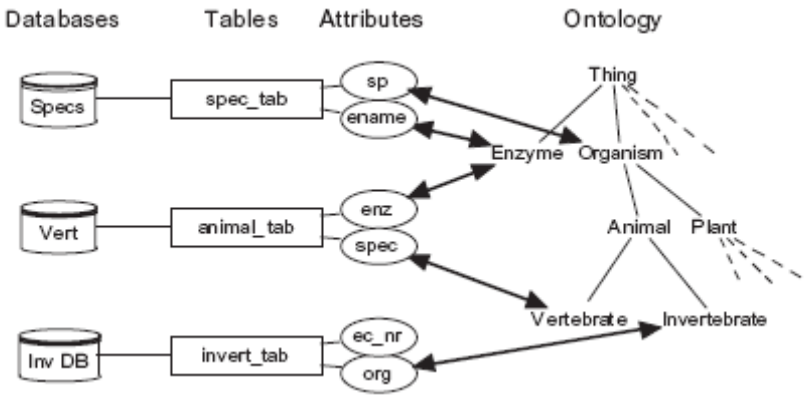
4.2. Χρήση Οντολογιών με Σκοπό τη Σημασιολογική Αναζήτηση σε Βάσεις Δεδομένων

Ένας επιπλέον τρόπος της χρήσης των οντολογιών είναι η αναζήτηση με σημασιολογικά κριτήρια σε βάσεις δεδομένων. Στην περίπτωση αυτή τα διάφορα συστατικά μιας βάσης (πίνακες, πεδία, και δεδομένα) αντιστοιχίζονται με κάποια concepts μιας οντολογίας, ώστε να μπορεί να υποστηριχθεί η αναζήτησή δεδομένων με βάση τη σημασία τους και όχι με βάση το ακριβές ταίριασμά τους με δεδομένα της βάσης. Επίσης, με τέτοιου είδους χρήσης οντολογιών ο χρήστης δε χρειάζεται να έχει εμπεριστατωμένη γνώση για τη δομή της εκάστοτε βάσης στην οποία θέλει να κάνει την αναζήτησή του.

Οι Jacob Kohler et. al [30] αναπτύσσουν ένα σύνολο αρχών για σημασιολογική ολοκλήρωση βάσεων δεδομένων, καθώς και ένα σύστημα για την εφαρμογή αυτών των αρχών, το SEMEDA (Semantic Meta Database), που ως στόχο έχει την αναζήτηση σε ποικίλες βάσεις δεδομένων βιοϊατρικής με σημασιολογικό τρόπο και χωρίς την ανάγκη γνώσης της δομής των

βάσεων που βρίσκονται πίσω από το σύστημα. Τέσσερα είναι τα βασικά προβλήματα που θεωρούν οι συγγραφείς πως υπάρχουν στο συγκεκριμένο τομέα: 1) Το γεγονός πως οι διαφορετικές βάσεις δεδομένων χρησιμοποιούν διαφορετικές λέξεις για το ίδιο πράγμα, το οποίο μπορεί να αντιμετωπιστεί με τη χρήση των controlled vocabularies ή των οντολογιών, όμως δεν υπάρχει κάποια συστηματική μέθοδος που να ορίζει ποια βάση δεδομένων χρησιμοποιεί ποιο controlled vocabulary. Έτσι ή δεν χρησιμοποιείται controlled vocabulary ή χρησιμοποιούνται διαφορετικά μεταξύ των διάφορων βάσεων. 2) Τα ονόματα των πεδίων των βάσεων δεδομένων, είτε δεν επεξήγονται σωστά, είτε είναι παραπλανητικά, είτε ισοδύναμα πεδία χρησιμοποιούν διαφορετικά ονόματα σε διαφορετικές βάσεις δεδομένων. 3) Η επερώτηση βάσεων δεδομένων απαιτεί γνώση για τα περιεχόμενα των πινάκων τους. Αυτό δεν αποτελεί τόσο μεγάλο πρόβλημα όταν χρησιμοποιείται η διαπροσωπεία της ίδιας της βάσης, αλλά το πρόβλημα επιδεινώνεται στα συστήματα ενσωμάτωσης βάσεων δεδομένων, που έχουν τη δική τους διαπροσωπεία επερωτήσεων. Στα συστήματα αυτά οι πίνακες της εκάστοτε βάσης θα πρέπει να επισημειωθούν σημασιολογικά με επιπλέον πληροφορία. 4) Λόγω έλλειψης συστηματικών μηχανισμών σύνδεσης μεταξύ βάσεων δεδομένων, σχεδόν όλα τα συστήματα ενσωμάτωσης βάσεων συνδέουν μόνο τα πιο σημαντικά πεδία τους. Η μέθοδος που προτείνεται αποτελείται από τα εξής βήματα:

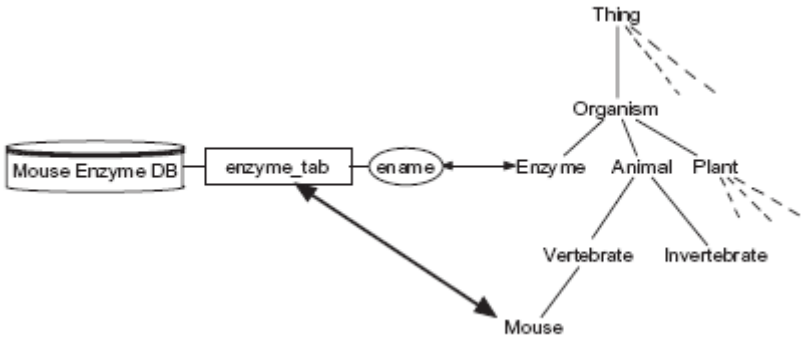
- **Σημασιολογική επισημείωση των πεδίων κάθε πίνακα:** Αντιστοιχίζονται τα κατάλληλα concepts της οντολογίας σε κάθε πεδίο πίνακα μιας βάσης



Σχήμα 4.5 Σημασιολογική Επισημείωση των Πεδίων ενός Πίνακα. [30]

Στο παράδειγμα του Σχήματος 4.5 το "ename" και το "enz" ορίζονται από το ίδιο concept "Enzym", έτσι δίνεται πως σημασιολογικά και τα δύο πεδία περιέχουν ονόματα ενζύμων. Το πεδίο "org" περιέχει ασπόνδυλους οργανισμούς, ενώ το "sp" περιέχει ζώα και φυτά.

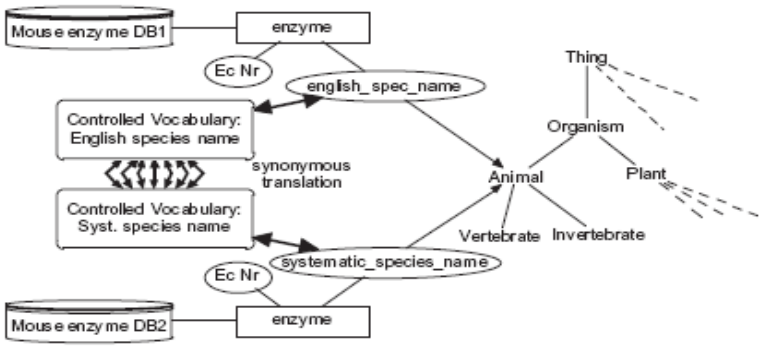
- **Σημασιολογική επισημείωση των πινάκων κάθε βάσης:** Αντιστοιχίζονται τα κατάλληλα concepts της οντολογίας σε κάθε πίνακα της βάσης. Με τη διαδικασία αυτή όλες οι καταχωρήσεις ενός πίνακα επισημειώνονται σημασιολογικά με τον ίδιο τρόπο.



Σχήμα 4.6 Σημασιολογική Επισημείωση των Πινάκων μιας Βάσης. [30]

Στο παράδειγμα του Σχήματος 4.6 τα περιεχόμενα του πίνακα "enzyme_tab" περιέχουν μόνο δεδομένα που αφορούν ποντίκια, αλλά δεν περιέχει κάποιο πεδίο που να περιέχει το είδος του οργανισμού στον οποίο αναφέρονται. Έτσι ο πίνακας επισημειώνεται για να υποδειχθεί πως περιέχει δεδομένα για ποντίκια.

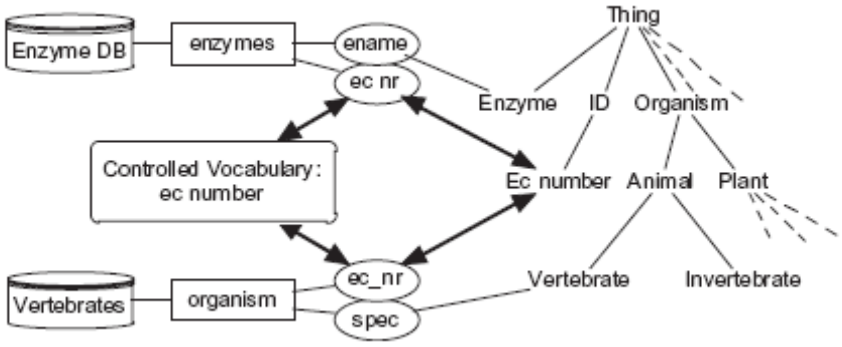
- **Σημασιολογική επισημείωση των τιμών που δέχονται τα πεδία του κάθε πίνακα:** Η σημασιολογία των τιμών ενός πεδίου, μπορεί να ορισθεί με τη χρήση controlled vocabularies, σαν τύπους (datatypes) των πεδίων, πράγμα που γίνεται σε αρκετές βάσεις.



Σχήμα 4.7 Σημασιολογική Επισημείωση των Τιμών ενός Πεδίου. [30]

Αντιστοιχίζοντας συνώνυμα concepts μεταξύ των controlled vocabularies είναι δυνατόν να συσχετιστούν δεδομένα που υπάρχουν στη βάση και χρησιμοποιούν διαφορετικούς όρους για το ίδιο πράγμα. Ένα παράδειγμα φαίνετε στο Σχήμα 4.7.

- **Σύνδεσμοι βάσεων και παραπομπές (cross-references):** Με την εφαρμογή των τριών προηγούμενων βημάτων μπορούν να προκύψουν παραπομπές για ένα πεδίο Α π.χ. το σύνολο όλων των πεδίων που έχουν κοινό σημασιολογικό ορισμό και χρησιμοποιούν κοινό controlled vocabulary. Με τις παραπομπές μπορεί να γίνει αυτόματη παραγωγή συνδέσεων βάσεων δεδομένων όπως επίσης να προκύψει ποιοι πίνακες μπορούν να κάνουν join μεταξύ τους. Ένα σύστημα, μπορεί έτσι, αυτόματα να κάνει σύνδεση σε άλλους πίνακες που περιέχουν επιπλέον πληροφορία για κάτι, ένα τέτοιο παράδειγμα φαίνεται στην Σχήμα 4.8.



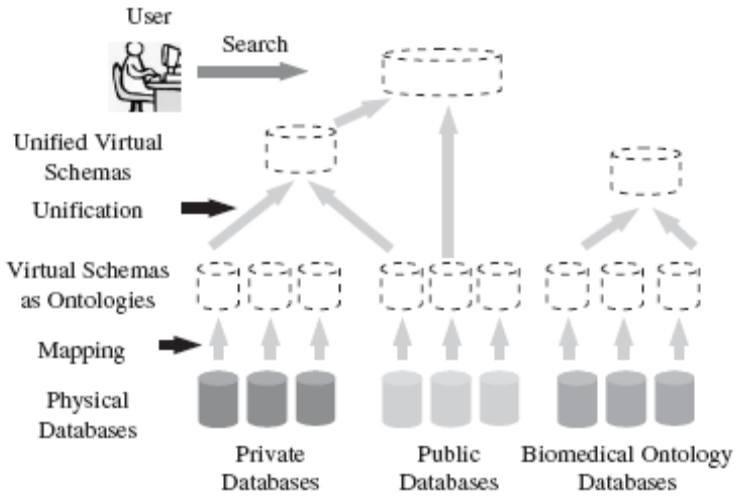
Σχήμα 4.8 Σύνδεσμοι Βάσεων και Παραπομπές. [30]

Στο Σχήμα 4.8 ένα ερώτημα για πεδίο “animal” και τιμή “mouse”, θα εντοπίσει τα “ec numbers” των ενζύμων του ποντικιού και χρησιμοποιώντας την παραπομπή, το σύστημα μπορεί να δημιουργήσει αυτόματα σύνδεσμο σε άλλους πίνακες βάσεων δεδομένων με επιπλέον πληροφορία για τα “ec numbers”, στο παράδειγμα της εικόνας θα βρει επιπρόσθετη πληροφορία από τη βάση “enzymes”.

Το σύστημα SEMEDA που υλοποιήσανε έχει μια αρχιτεκτονική τριών επιπέδων, όπου στο πίσω μέρος βρίσκεται μια βάση δεδομένων που κρατά τις οντολογίες, τα metadata των βάσεων και τους σημασιολογικούς ορισμούς, και στο ενδιάμεσο χρησιμοποιείται java για τη δυναμική παραγωγή της HTML που βρίσκεται στο εμπρός επίπεδο. Το SEMEDA μπορεί να χειριστεί μεγάλα controlled vocabularies και οντολογίες με πολύ μεγάλο πλήθος από concepts. Χρησιμοποιεί μια δική του οντολογία για ενσωμάτωση βάσεων. Η οντολογία αυτή είναι μια μικρή top-ontology που ορίζει τις βάσεις δεδομένων σε επίπεδο σχήματος. Σε αντίθεση οντολογίες όπως η Gene Ontology, χρησιμοποιούνται ως controlled vocabularies που χρησιμοποιούνται για την ενοποίηση των δεδομένων μεταξύ διαφορετικών βάσεων. Λόγω του ότι οι βάσεις μοριακής βιολογίας γενικά αποθηκεύουν “names”, identifiers”, “properties” και “free text description” , αυτά έχουν συμπεριληφθεί ως concepts (name, identifier, description, property) στην top-level οντολογία του SEMEDA. Το σύστημα προσφέρει τρία γραφικά περιβάλλοντα στους χρήστες το Meta DB που προσφέρει την εξερεύνηση και την επεξεργασία για metadata, οντολογίες και σημασιολογικούς ορισμούς, το Admin tool για λειτουργίες διαχείρισης και το Query DB από όπου ο χρήστης μπορεί να δώσει τα ερωτήματά του και να πάρει τις κατάλληλες απαντήσεις.

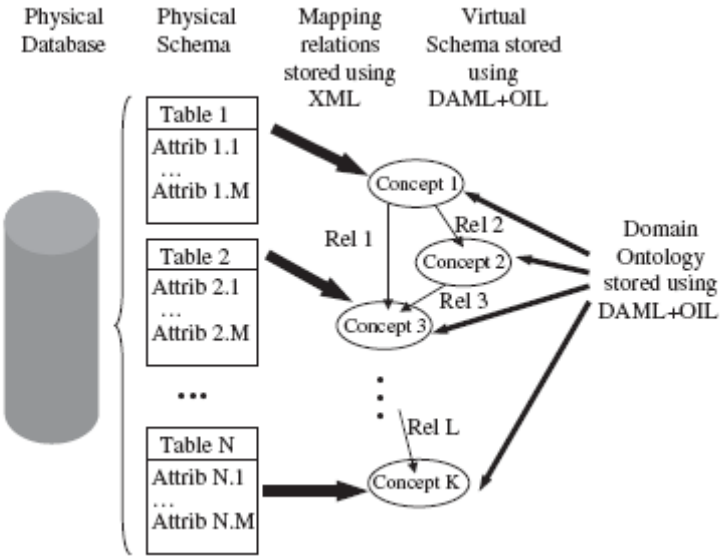
Στην εργασία των D. Perez-Rey et al. [31], αναπτύχθηκε ένα σύστημα, το ONTOFUSION, βασισμένο σε οντολογίες με σκοπό την ενοποίηση ετερογενών βιοϊατρικών βάσεων δεδομένων. Η ενοποίηση βάσεων δεδομένων απαιτεί την γεφύρωση των συντακτικών και σημασιολογικών διαφορών που υφίστανται μεταξύ διαφορετικών πηγών δεδομένων, ένα πρόβλημα στο οποίο οι οντολογίες είναι ιδανικές για την επίλυσή του. Στα συστήματα αυτά, χρησιμοποιούνται όψεις των βάσεων δεδομένων, βασισμένες σε οντολογίες, για τη διευκόλυνση της αντιστοίχησης των αντικειμένων που ανήκουν σε μια συγκεκριμένη βάση δεδομένων σε έννοιες ενός διαμοιραζόμενου λεξικού. Αν δύο διαφορετικές βάσεις δεδομένων εμπεριέχουν την ίδια έννοια, αλλά αυτή αναπαρίσταται με διαφορετικά ονόματα, οι οντολογίες χρησιμοποιούνται στην αντιστοίχηση αυτών των ονομάτων στην κοινή έννοια.

Η ενοποίηση βάσεων δεδομένων με βάση τη σημασιολογία είναι βασική προϋπόθεση για παροχή ομογενούς πρόσβασης σε κλινικές και γενετικές βάσεις. Η προσέγγιση που χρησιμοποιήθηκε από το ONTOFUSION για ενοποίηση είναι βασισμένη σε δύο λειτουργίες: την αντιστοίχηση (mapping) και την ενοποίηση (unification). Στη διαδικασία αντιστοίχησης , το φυσικό σχήμα κάθε βάσης αντιστοιχίζεται σε αυτό που οι συγγραφείς ονομάζουν "εικονικό σχήμα (virtual schema) ". Τα εικονικά σχήματα είναι οντολογίες που αναπαριστούν τη δομή της πληροφορίας που περιέχεται σε μια βάση, σε ένα εννοιολογικό επίπεδο. Στη διαδικασία ενοποίησης, περισσότερα του ενός εικονικά σχήματα, που το καθένα αντιστοιχεί σε διαφορετική βάση δεδομένων, ενώνονται σε ένα ενοποιημένο εικονικό σχήμα. Τα ενοποιημένα εικονικά σχήματα είναι οντολογίες που αποδίδουν την εννοιολογική δομή της πληροφορίας που είναι αποθηκευμένη σε ποικίλες βάσεις. Έτσι ενεργούν σαν περιγραφείς εικονικών βάσεων, που κάνουν ταίριασμα πραγματικών δεδομένων από φυσικές βάσεις. Στην Σχήμα 4.9 περιγράφονται οι διαδικασίες που ακολουθούνται από το ONTOFUSION. Το πρώτο επίπεδο από κάτω δείχνει τα τρία διαφορετικά είδη φυσικών βάσεων που χρησιμοποιήθηκαν στο ONTOFUSION. Στο δεύτερο επίπεδο φαίνεται η αντιστοίχηση των φυσικών βάσεων σε εικονικά σχήματα. Τέλος στο τρίτο επίπεδο φαίνεται η διαδικασία ενοποίησης πολλών εικονικών σχημάτων σε ενοποιημένα εικονικά σχήματα, τα οποία μπορούν να χρησιμοποιηθούν από χρήστες για να ανακτήσουν πληροφορίες από ποικίλες πηγές ταυτόχρονα.



Σχήμα 4.9 Οι Διαδικασίες που Ακολουθούνται στο ONTOFUSION. [31]

Για τη δημιουργία των εικονικών σχημάτων χρησιμοποιούνται οντολογίες πεδίου, οι οποίες έχουν ως σκοπό την παροχή ενός κοινού λεξικού, που εγγυάται πως μόνο γενικά αποδεκτά ονόματα εννοιών χρησιμοποιούνται για την περιγραφή των αντικειμένων των εικονικών σχημάτων (Σχήμα 4.10). Οι οντολογίες είναι αυτές που σιγουρεύουν πως σημασιολογικά ισοδύναμα αντικείμενα σε διαφορετικά σχήματα θα αντιστοιχηθούν στην ίδια έννοια. Η διαδικασία αυτή είναι ημιαυτόματη και οι συγγραφείς δημιούργησαν μια εφαρμογή για την ολοκλήρωσή της.



Σχήμα 4.10 Χρήση Οντολογιών Πεδίου για τη Δημιουργία Εικονικών Σχημάτων. [31]

Κατά τη διαδικασία ενοποίησης των εικονικών σχημάτων, ο αλγόριθμος ενοποίησης, ελέγχει την οντολογία που χρησιμοποιήθηκε και βρίσκει ποιες έννοιες από τα εικονικά σχήματα πρέπει να ενοποιηθούν. Όταν δύο ή περισσότεροι όροι ταιριάζουν με την ίδια έννοια στην οντολογία, τότε ενοποιούνται σε μία έννοια. Η διαδικασία αυτή γίνεται τελείως αυτόματα. Τέλος δημιούργησαν μια εφαρμογή που βοηθά το χρήστη να επιλέγει τις βάσεις από τις οποίες θέλει να ανακτήσει πληροφορίες, και με πλοήγηση στο εικονικό σχήμα να επιλέγει έννοιες για τις οποίες θέλει συγκεκριμένες πληροφορίες από τις βάσεις. Έπειτα το σύστημα δημιουργεί τα κατάλληλα ερωτήματα για την κάθε βάση, ώστε να μπορέσει να ανακτήσει τις πληροφορίες από τις φυσικές βάσεις και να τις επιστρέψει στον χρήστη.

Οι Benslimane et al. στο [40] δίνουν μια προσέγγιση (terminology-based) για την ανάπτυξη οντολογιών που εκμεταλλεύεται υπάρχουσες οντολογικές πηγές από ταξινομίες και σχήματα βάσεων δεδομένων.

Τονίζεται η σπουδαιότητα ύπαρξης συγκεκριμένων ορολογιών που είναι οργανωμένες σε ταξινομίες και της ύπαρξης πληθώρας βάσεων δεδομένων. Ταξινομία είναι μια ιεραρχία εννοιών με συνδέσμους του τύπου is-a και part-of και μπορούν να θεωρηθούν ένα είδος απλοποιημένων οντολογιών που δεν περιέχουν πλήρη αξιώματα και ορισμούς των όρων που περιλαμβάνουν. Υπάρχουν πολλές τέτοιες ταξινομίες σε πολλά πεδία όπως και στο πεδίο της ιατρικής. Έτσι, τέτοιες δομημένες ορολογίες είναι πολύ χρήσιμες, αφού παρέχουν ένα διαμοιραζόμενο λεξικό, αλλά είναι ελλιπείς σε επιπλέον πληροφορία που παραμένει αυτονόητη. Επίσης, η πληθώρα των δεδομένων που είναι συσσωρευμένα σε αντίστοιχο πλήθος βάσεων δεδομένων είναι μια πολύ καλή πηγή για τη δημιουργία οντολογιών. Όμως οι βάσεις δεδομένων υποφέρουν από την έλλειψη σημασιολογίας και είναι περιορισμένες στη χρήση τους σε συγκεκριμένες εφαρμογές. Οι οντολογίες παρέχουν μια διαμοιραζόμενη κατανόηση που υπόσχεται την επίλυση σημασιολογικών αντιφάσεων. Ένας ημιαυτόματος μετασχηματισμός των σχέσεων των βάσεων δεδομένων σε οντολογίες είναι εφικτός, και κάθε οντολογία από αυτές μπορεί να επιτρέπει τον διαμοιρασμό και διακίνηση της πληροφορίας.

Τα βήματα που προτείνουν είναι τα εξής:

- Επιλογή της ορολογίας: η εύρεση και ομοφωνία για όρους που είναι πολύ αντιπροσωπευτικοί για στο πεδίο εφαρμογής. Οι ταξινομίες αυτές χαρακτηρίζονται από αντιφάσεις και σημασιολογική ανομοιογένεια.
- Συγχώνευση των ταξινομιών: σε αυτό το βήμα γίνεται μια ανάλυση των ταξινομιών για να προσδιοριστούν οι αντιφάσεις και η σημασιολογική ανομοιογένεια τους.
- Διαδικασία ενοποίησης των σχημάτων των βάσεων δεδομένων: τα σχήματα των βάσεων παρουσιάζουν συντακτική, δομική και σημασιολογική ανομοιογένεια. Τέτοιες ανομοιογένειες επιλύονται με τη δημιουργία ενός γενικού/ενοποιημένου σχήματος που συνδυάζει τις πληροφορίες των διαφορετικών σχημάτων. Τα βήματα 2 και 3 είναι ανεξάρτητα οπότε μπορούν να τρέχουν παράλληλα.
- Σημασιολογικός εμπλουτισμός (semantic enrichment): οι έννοιες (concepts) των ενοποιημένων σχημάτων δεν εκφράζουν απαραίτητα το κοινό λεξιλόγιο. Έτσι ο

συνδυασμός της γενικής ταξινομίας και του γενικού σχήματος θα ευθυγραμμιστούν (alignment) για να παραχθεί η γενική οντολογία.

- Υλοποίηση: η οντολογία που προέκυψε από το προηγούμενο βήμα θα περιγραφεί σε μια τυπική γλώσσα (formal language).

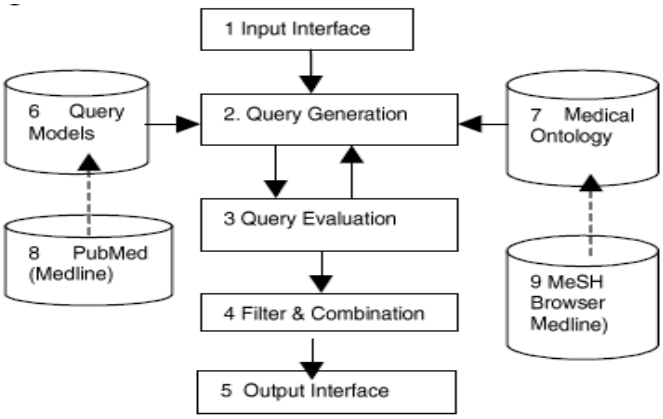
4.3. Χρήση Οντολογιών με Σκοπό την Ανάκτηση Πληροφορίας και Εγγράφων

Ένα ακόμη σημείο της χρήσης των οντολογιών στην ιατρική είναι η ανάκτηση πληροφορίας και εγγράφων από διάφορες και ετερογενείς πηγές, όπως βάσεις που περιέχουν δημοσιεύσεις πάνω στο συγκεκριμένο πεδίο. Με τις οντολογίες μπορούμε να κερδίσουμε στο ότι πλέον τις αναζητήσεις των κειμένων μπορούμε να τις πραγματοποιήσουμε εκμεταλλευόμενοι τη σημασιολογία των ερωτημάτων των χρηστών, καθώς και των ίδιων των κειμένων και όχι απλά βασιζόμενοι στην απλή εύρεση λέξεων κλειδιών μέσα σε κείμενα ή απλή στατιστική ανάλυση των κειμένων.

Ακολουθούν κάποιες σχετικές εργασίες στο συγκεκριμένο πεδίο.

Στην εργασία των Jose Maria Abasolo και Mario Gomez [32], γίνεται ανάπτυξη ενός βασισμένου σε οντολογίες agent για την ανάκτηση πληροφορίας. Σχεδιάσανε ένα σύστημα που αποτελείται από αρκετά δομικά στοιχεία (modular system), που μπορεί εύκολα να προσαρμοστεί σε διαφορετικές πηγές ιατρικής βιβλιογραφίας, ή διαφορετικά πεδία ενδιαφέροντος. Ο στόχος τους είναι η σχεδίαση μιας αρχιτεκτονικής τριών επιπέδων, η χρήση διαφορετικών οντολογιών και μοντέλων ερωτήσεων, και ο ορισμός τελεστών συνάθροισης για τον συνδυασμό αποτελεσμάτων από διαφορετικά ερωτήματα. Με το σύστημά τους θέλουν να επιλύσουν το πρόβλημα που παρουσιάζεται στους χρήστες που θέλουν να ανακτήσουν πληροφορία σε ένα εξειδικευμένο πεδίο, του να επαναδιατυπώνουν τα ερωτήματά τους μέχρι να επιτύχουν μια αποτελεσματική ανάκτηση που επιστρέφει κείμενα τα οποία έχουν απόλυτη σχέση με την πληροφορία που αναζητούν. Η αρχιτεκτονική του συστήματος φαίνεται στο Σχήμα 4.11. Παρακάτω δίνουμε μια περιγραφή για το εκάστοτε συστατικό της αρχιτεκτονικής:

- **Input interface:** επιτρέπει στο χρήστη να ορίσει τα keywords με τα οποία θα εκτελεστεί η αναζήτηση, καθώς ποιες ιατρικές κατηγορίες τον ενδιαφέρουν και κάποιους modifiers για να περιορίσει την αναζήτηση π.χ. ημερομηνία της δημοσίευσης. Το ερώτημα που σχηματίζεται με αυτά τα δεδομένα το ονομάζουν "Consultation", και είναι ένα πολύ αφηρημένο, ανεξάρτητο της βάσης δεδομένων και της οντολογίας υψηλού επιπέδου ερώτημα.



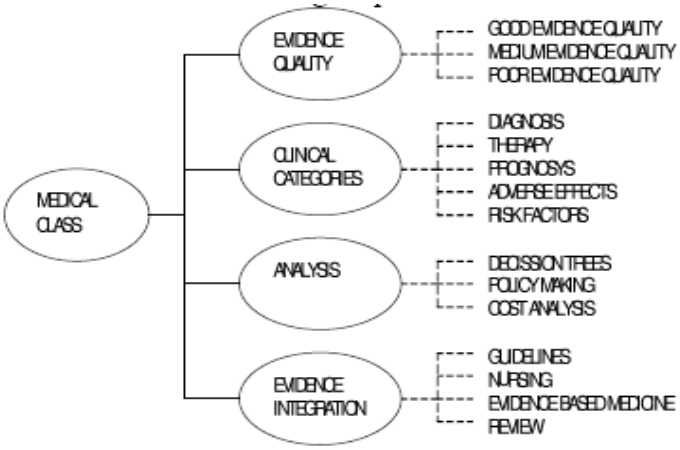
Σχήμα 4.11 Αρχιτεκτονική Συστήματος Ανάκτηση Εγγράφων. [32]

- **Query Generation & Reformulation:** Ένα consultation αποτελεί την είσοδο του Query Generator. Το συστατικό αυτό είναι ο πυρήνας του συστήματος. Χρησιμοποιεί το consultation, την ιατρική οντολογία και τα μοντέλα ερωτήσεων, έτσι ώστε να μετασχηματιστεί το consultation σε μια συλλογή από χαμηλού επιπέδου ερωτήματα, άμεσα εξαρτώμενα από την βάση δεδομένων. Τα χαμηλού επιπέδου ερωτήματα τα ονομάζουν "Specific Queries". Στο σημείο αυτό εμπλέκεται και ένα άλλο επίπεδο μεταξύ του consultation και των Specific Queries", που είναι βασιζόμενο σε ένα μοντέλο γνώσης, και το ονομάζουν "Conceptual Query", που είναι υπεύθυνο για τη μετατροπή των consultations σε specific queries. Επιπλέον το ενδιάμεσο αυτό επίπεδο μπορεί να επαναπροσδιορίσει τα Specific Queries όταν τα αποτελέσματα που επιφέρουν δεν είναι ικανοποιητικά.
- **Query Evaluation:** Τα αποτελέσματα που προκύπτουν από τα Specific Queries – μια συλλογή από βιβλιογραφικές αναφορές – περνάνε ως είσοδος στον query evaluator. Το συστατικό αυτό αποδίδει κάποια βαθμολογία στις αναφορές αυτές, σύμφωνα με κάποια κριτήρια, όπως τον βαθμό με τον οποίο ικανοποιούν τις απαιτήσεις του

χρήστη ή την ποιότητα των αποδείξεων των μελετών που αναφέρονται στη βιβλιογραφία. Επιπλέον τα αποτελέσματα των specific queries ομαδοποιούνται για το κάθε conceptual query.

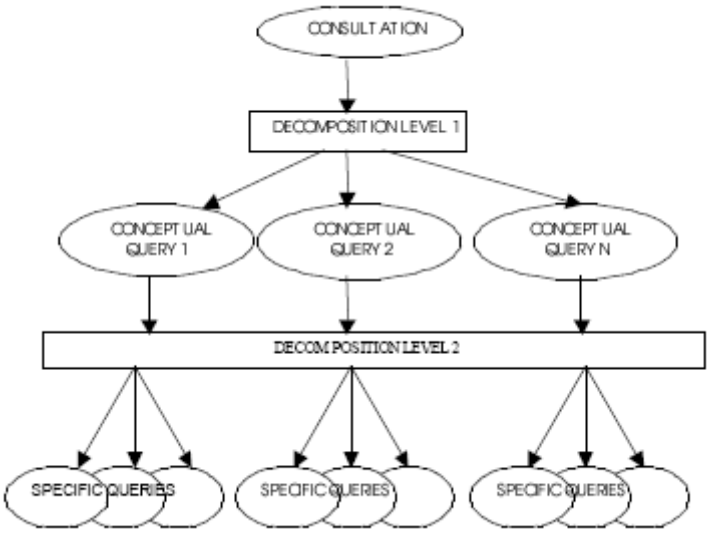
- **Filter & Combination:** Οι αναφορές της βιβλιογραφίας πρέπει να φιλτραριστούν και να συνδυαστούν για να πάρουν έναν τελικό βαθμό. Στο σημείο αυτό απομακρύνονται τα διπλότυπα και διαγράφονται αναφορές που δεν πληρούν κάποια ελάχιστα κριτήρια των αναζητήσεων των χρηστών.
- **Output Interface:** Στο σημείο αυτό τα αποτελέσματα επανασυνδυάζονται για να παρουσιαστούν στο χρήστη με κατάλληλο τρόπο.
- **Query Models:** Αναφέρονται σε σχήματα πληροφορίας που αναπαριστούν ερωτήματα σε διαφορετικά επίπεδα. Αυτά είναι τα consultations, conceptual queries και specific queries. Το συστατικό αυτό επιτρέπει το σύστημα να είναι όσο το δυνατόν ανεξάρτητο από τα συμφραζόμενα, έχοντας σαν συμφραζόμενα στην περίπτωση αυτή μια συγκεκριμένη μηχανή αναζήτησης όπως το PubMed.
- **Medical Ontology:** Περιέχει ιατρική πληροφορία που χρησιμοποιείται για την παραγωγή των ερωτημάτων. Οι συγγραφείς χρησιμοποιούν μια οντολογία που δημιούργησαν οι ίδιοι και που βασίζεται στο Mesh.
- **PubMed & Medline:** Στη συγκεκριμένη φάση οι συγγραφείς χρησιμοποιούν μόνο μια βάση δεδομένων, τη Medline, και την μηχανή αναζήτησης που την υποστηρίζει, το PubMed.
- **MeSH Browser:** Παρέχει την πρόσβαση στην οντολογία MeSH, και επιτρέπει στο χρήστη να παίρνει κάποιες επιπλέον πληροφορίες για τα keyword που χρησιμοποιεί.

Στο Σχήμα 4.12 δίνεται η δομή της οντολογίας που χρησιμοποιείται. Είναι οργανωμένη σε τρία επίπεδα, ένα που έχει βασικές περιοχές ιατρικών θεμάτων, ένα που περιέχει μια συλλογή από ιατρικές κατηγορίες και η κάθε μία από αυτές στο τρίτο επίπεδο περιέχει όρους από τη MeSH οντολογία.



Σχήμα 4.12 Η Δομή της Οντολογίας. [32]

Επιπλέον στο Σχήμα 4.13 δίνετε μια σχηματική αναπαράσταση του τρόπου με τον οποίο ένα ερώτημα από την consultation μορφή του αποσυντίθεται σε περισσότερα specific queries. Η αντίστροφη σειρά ακολουθείται όταν έχουν βγει τα αποτελέσματα από κάθε specific query, έχουν βαθμολογηθεί οι αναφορές και μετά ομαδοποιούνται με τέτοιο τρόπο ώστε να παρουσιαστούν στο χρήστη. Αφού δοθούν τα keywords, οι modifiers και οι ιατρικές κατηγορίες που ενδιαφέρουν τον χρήστη στην consultation μορφή, για κάθε κατηγορία δημιουργείται ένα conceptual query, το οποίο με τη σειρά του αποσυντίθεται σε πολλά specific queries. Τα specific queries δημιουργούνται με το συνδυασμό των keywords, και κάθε όρου που υπάρχει στην οντολογία στη συγκεκριμένη κατηγορία για την οποία δημιουργήθηκε το conceptual query καθώς και των modifiers. Αυτά με τη σειρά τους δίνονται ως είσοδο στη μηχανή αναζήτησης και λαμβάνονται οι απαντήσεις. Οι αναφορές της βιβλιογραφίας που επιλέχθηκαν ως απαντήσεις περνάν στο στάδιο του evaluation, όπου παίρνουν μια βαθμολογία ανάλογα με το πόσες φορές επιστράφηκαν ως απάντηση στα specific queries ενός conceptual query. Στη συνέχεια περνάν στο στάδιο του combination όπου χρησιμοποιείται μια συνάρτηση συνάθροισης που δίνει μια τελική βαθμολογία σε κάθε αναφορά, συνδυάζοντας πλέον τις βαθμολογίες που ελήφθησαν στα διαφορετικά conceptual queries. Έτσι πλέον μπορούν να παρουσιαστούν ταξινομημένα ως έξοδος στον χρήστη.



Σχήμα 4.13 Σχηματική Αναπαράσταση Μετατροπής Consultation Query σε Specific Query. [32]

Οι Jimmy Lin et. al [33] πραγματεύονται την ανάπτυξη ενός πλαισίου εργασίας μέσω του οποίου εντοπίζουν τα είδη της γνώσης που είναι σημαντικά στο πεδίο της αναζήτησης πληροφορίας. Στη συνέχεια εφαρμόζουν αυτό το πλαίσιο εργασίας στο πεδίο της κλινικής ιατρικής, και πιο συγκεκριμένα σε ένα πρακτικό πεδίο της που ονομάζεται evidence-based medicine. Τα τρία είδη γνώσης που οι συγγραφείς θεωρούν απαραίτητα για την σημασιολογική αναζήτηση γνώσης είναι τα εξής:

- **Γνώση που έχει να κάνει με τη δομή του προβλήματος (knowledge about the problem structure):** Τι αναπαραστάσεις είναι χρήσιμες για τη σύλληψη της γνώσεις που χρειαζόμαστε. Οι αναπαραστάσεις αυτές μπορεί να αφορούν νοητικές δομές ενός ειδικού του πεδίου (π.χ. ο τρόπος με τον οποίο αποσυνθέτει το πρόβλημα και αναλύει τα αποτελέσματα που ανακτήθηκαν) ή μπορεί να αφορούν κάποια υπολογιζόμενα στοιχεία (ή και τα δύο).
- **Γνώση που αφορά τις εργασίες των χρηστών (Knowledge about user tasks):** Πληροφορία που έχει να κάνει με το λόγο για τον οποίο κάποια πληροφορία είναι αναγκαία και πως θα αξιοποιηθεί αργότερα. Συνήθως η αναζήτηση κάποιας πληροφορίας είναι το αρχικό σημείο για κάποια άλλη δραστηριότητα (π.χ. η σύνταξη μιας αναφοράς, η λήψη μιας απόφασης κ.α.).
- **Γνώση για το πεδίο του ενδιαφέροντος (Knowledge about the domain):** Τι υπόβαθρό γνώσης για το συγκεκριμένο πεδίο είναι αναγκαίο να κατέχει ο χρήστης

μιας αναζήτησης για να μπορέσει να διατυπώσει ερωτήσεις και να ερμηνεύσει τις αντίστοιχες απαντήσεις. Εδώ περιλαμβάνεται γνώση για όρους που χρησιμοποιούνται για την αναπαράσταση εννοιών, καθώς και σχέσεων μεταξύ των εννοιών.

Τα είδη αυτά γνώσης, οι συγγραφείς τα θέτουν σαν ένα πλαίσιο εργασίας, που αν υπάρχουν σε έναν τομέα, τότε μπορεί να πραγματοποιηθεί μια διαδικασία σημασιολογικής ενοποίησης μεταξύ αναπαραστάσεων που κωδικοποιούν τις ανάγκες των χρηστών για πληροφορία και αντίστοιχων αναπαραστάσεων που αυτόματα εξάγονται από συλλογές κειμένων. Την ιδέα αυτή την συγκεκριμενοποιούν σε ένα συγκεκριμένο πεδίο, αυτό της κλινικής ιατρικής. Πιο συγκεκριμένα παίρνουν ένα συγκεκριμένο παράδειγμα πρακτικής της κλινικής ιατρικής, που ονομάζεται evidence-based medicine (EBM). Στο EBM καθορίζονται τρεις δομές γνώσης, που αν αξιοποιηθούν ταυτόχρονα περιγράφουν ένα μοντέλο που μπορεί να περιγράψει τις πολύπλοκες ανάγκες της κλινικής πληροφορίας. Αυτές είναι οι εξής:

- **Clinical Tasks:** Therapy, Diagnosis, Prognosis, Etiology
- **PICO elements:** Problem/Population, Intervention, Comparison, Outcome
- **Strength of Evidence:** Που περιλαμβάνει τρία επίπεδα, A-level evidence, B-level evidence και C-Level evidence

Αυτές οι τρεις δομές γνώσης παρέχουν τους δύο από τους τρεις τύπους γνώσης που είναι απαραίτητοι στην εννοιολογική ανάκτηση πληροφορίας. Οι Clinical Tasks δίνουν πληροφορία για τις δραστηριότητες των χρηστών και η δομή Strength of Evidence εξετάζει την ορθότητα των κλινικών συμφραζομένων σύμφωνα με τις δραστηριότητες αυτές. Η PICO αναπαράσταση παρέχει γνώση για την δομή του προβλήματος, για τη σύλληψη των αναγκών για κλινική πληροφορία. Επιπλέον για το τρίτο είδος γνώσης (knowledge about the domain) μπορεί να χρησιμοποιηθεί η οντολογία του UMLS. Έτσι πλέον μπορεί να γίνει μια σημασιολογική ενοποίηση μεταξύ των αναγκών των χρηστών που είναι εκφρασμένες σύμφωνα με την περιγραφή του PICO και αντίστοιχων δομών που εξάγονται από τα abstract του MEDLINE. Αυτή η διαδικασία ταιριάσματος πρέπει να είναι σύμφωνη με τις αντίστοιχες Clinical Tasks. Έτσι για παράδειγμα η ερώτηση “In Children with an acute febrile illness, what is the efficacy of single medication therapy with acetaminophen or ibuprofen in reducing fever?” μπορεί να αναπαρασταθεί ως:

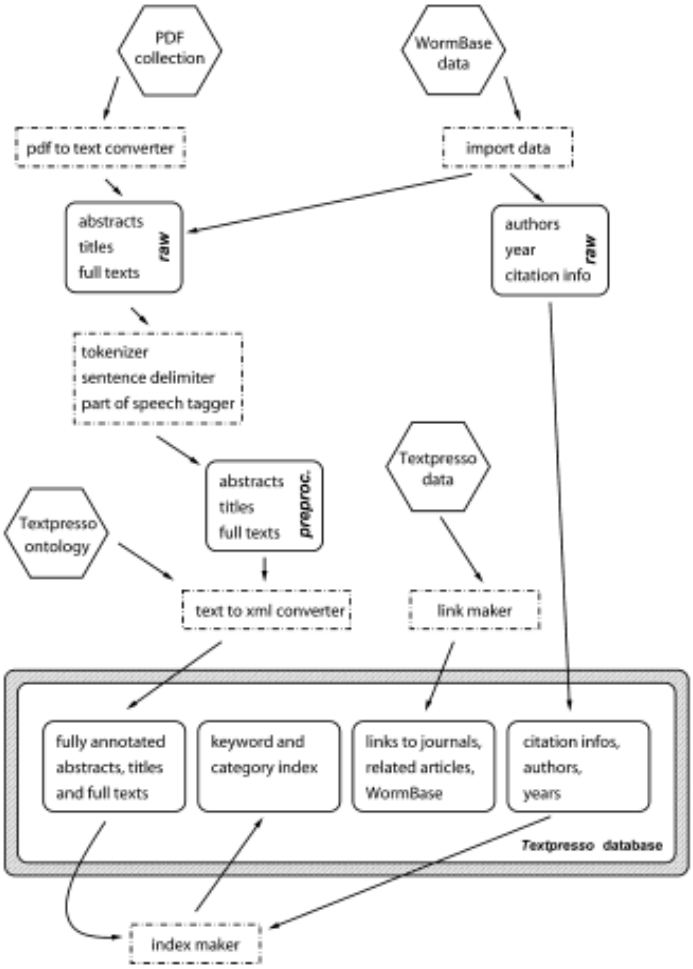
- **Task**: therapy
- **Problem**: acute febrile illness
- **Population**: children
- **Intervention**: acetaminophen
- **Comparison**: ibuprofen
- **Outcome**: reducing fever

Στη συνέχεια αφού γίνει επεξεργασία των abstracts του MEDLINE και εξαχθούν αντίστοιχα PICO δομικά στοιχεία από αυτά, μπορεί να γίνει ένα ταίριασμα και να δοθεί μια κατάλληλη απάντηση στον χρήστη. Η όλη αυτή διαδικασία βέβαια χρειάζεται αρκετά συστατικά για να πραγματοποιηθεί: ένα συστατικό για εξαγωγή γνώσης που αυτόματα θα εντοπίζει PICO δομικά στοιχεία στα abstracts του MEDLINE, έναν αλγόριθμο σκοραρίσματος για την καταλληλότητα ενός abstract σύμφωνα με την evidence-based medicine, και ένα συστατικό σύνθεσης απαντήσεων.

Η συγκεκριμένη εργασία επικεντρώνεται στο δεύτερο συστατικό αυτής της διαδικασίας. Οι συγγραφείς δίνουν έναν αλγόριθμο σκοραρίσματος που αντικατοπτρίζεται από τη συνάρτηση: $S_{EBM}= \lambda_1 S_{PICO} + \lambda_2 S_{SoE} + (1- \lambda_1- \lambda_2) S_{MeSH}$. Στη συνάρτηση αυτή το πρώτο συστατικό υπολογίζεται ανάλογα με το κατά πόσο ένα abstract είναι κατάλληλο για ένα ερώτημα εκφρασμένο σε μια PICO αναπαράσταση, το δεύτερο ανάλογα με την ποιότητα του περιοδικού ή του συνεδρίου που είναι δημοσιευμένο, τον τύπο της μελέτης και την παλαιότητά της, και το τρίτο ανάλογα με το κατά πόσο το abstract είναι σύμφωνο με τις user tasks με τις οποίες συσχετίζεται το ερώτημα. Έτσι πλέον με τον αλγόριθμο αυτό μπορεί να εντοπιστεί το κατά πόσο βοηθάνε οι τύποι γνώσης του πλαισίου εργασίας που δόθηκε από τους συγγραφείς στην ανάκτηση πληροφορίας, συγκρίνοντάς το με άλλες τεχνικές σκοραρίσματος, που είναι βασισμένες σε term matching και δεν χρησιμοποιούν σημασιολογία. Οι αξιολόγηση των αποτελεσμάτων που δίνουν δείχνει πως υπάρχει μεγάλη βελτίωση χρησιμοποιώντας αυτούς τους τύπους γνώσης.

Οι Hans-Michael Muller et. al [34] αναπτύξανε ένα σύστημα για σημασιολογική ανάκτηση κειμένων και αναζήτησης πληροφορίας στον τομέα της βιοϊατρικής, το Textpresso. Οι ερευνητές αναπτύξανε μια οντολογία που απαρτίζεται από τρία συστατικά, κλάσεις που

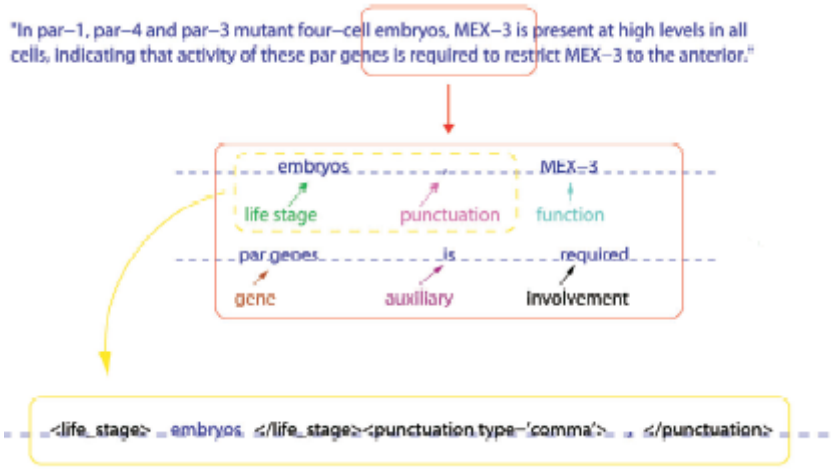
αντικατοπτρίζουν βιοϊτρικές έννοιες, κλάσεις που περιγράφουν σχέσεις μεταξύ των βιοϊατρικών εννοιών και τις μεταξύ τους συσχετίσεις, και τέλος κλάσεις που εμπεριέχουν συντακτικούς και γραμματικούς κανόνες. Οι όροι κάποιον κλάσεων της πρώτης κατηγορίας περιγράφηκαν με PERL κανονικές εκφράσεις, για να μπορέσουν να ταιριάξουν λέξεις ή φράσεις που μοιάζουν στη γραφή με όρους αυτών των κλάσεων (π.χ. στην κλάση "gene" περιλαμβάνουν την κανονική έκφραση [A-Za-z][a-z][a-z]-\d+ για να μπορέσουν να συλλάβουν τα γονίδια που γράφονται με τρία γράμματα του λατινικού αλφαβήτου, ακολουθώντας τα μια παύλα και ένα αριθμός, όπως το let-60). Επίσης αρκετές κλάσεις έχουν και μια σειρά από υποκατηγορίες κλάσεων, που βοηθούν σε μια πιο εξεζητημένη αναζήτηση. Η οντολογία τους απαρτίζεται από 33 βασικές κλάσεις. Εστιάσανε στην εφαρμογή του μοντέλου τους σε κείμενα που έχουν να κάνουν με το "Caenorhabditis elegans" που είναι ένα είδος νηματόζωου. Επιλέξανε το συγκεκριμένο τομέα, γιατί η βιβλιογραφία του είναι σχετικά μικρή ώστε να μπορέσει να συλλεχθεί, να είναι ολοκληρωμένη και να απαρτίζεται από ολόκληρα τα κείμενα και όχι μέρος τους (π.χ. τα abstracts). Το λεξιλόγιο της οντολογίας εμπλουτίστηκε από όρους της Gene Ontology (GO), καθώς και από όρους που βρέθηκαν σε διάφορες βάσεις δεδομένων, όπως την WormBase. Οι συγγραφείς επίσης δημιούργησαν μόνοι τους ένα λεξικό ονομάτων σχετικών με τον συγκεκριμένο τομέα, αφού αποδείχθηκε παλιότερα πως μια αυτόματη διαδικασία αναγνώρισης ονομάτων σχετικών με το πεδίο της βιολογίας είναι δύσκολο να πραγματοποιηθεί. Σε αυτή τους την προσπάθεια βοηθήθηκαν από το γεγονός πως στο συγκεκριμένο πεδίο που ασχολείται η εργασία τους, (το C. Elegans), οι επιστήμονες χρησιμοποιούν μια συγκεκριμένη πειθαρχία στην επιλογή των ονομάτων, καθώς και το ότι η WormBase περιέχει ένα σύνολο αυτών. Στο Σχήμα 4.14 φαίνεται ο τρόπος λειτουργίας του συστήματός τους.



Σχήμα 4.14 Ο Τρόπος Λειτουργίας του Textpresso. [34]

- Αρχικά συλλέγονται τα κείμενα του τομέα σε PDF μορφή
- Επιπλέον βιβλιογραφικά στοιχεία συλλέγονται από τη WormBase
- Τα δεδομένα που συλλέχθηκαν από τις παραπάνω πηγές αντιμετωπίσθηκαν με διαφορετικό τρόπο
    - Ο συγγραφέας, η χρονολογία δημοσίευσης και οι διάφορες πληροφορίες της δημοσίευσης, εισήχθησαν όπως είναι στη βάση δεδομένων
    - Τα abstracts, οι τίτλοι και το σώμα των εγγράφων δέχθηκαν περαιτέρω επεξεργασία
- Αρχικά έγινε tokenization στα κείμενα, χωρίζοντάς τα σε προτάσεις

- Στη συνέχεια γίνεται χρήση ενός part-of speech tagger, που αντιστοιχίζει ένα γραμματικό tag σε κάθε λέξη του κειμένου. Τα tag αυτά δε χρησιμοποιήθηκαν στη συγκεκριμένη έκδοση του Textpresso
- Μετά από την παραπάνω προεπεξεργασία γίνεται χρήση της οντολογίας , ώστε να εντοπιστούν και να επισημειωθούν με tags οι τίτλοι, τα abstracts και ολόκληρο το κείμενο των εγγράφων. Τα tags αυτά περιέχουν τα ονόματα των κατηγοριών της οντολογίας με τις οποίες συσχετίζονται τα σημεία του κειμένου, καθώς και ιδιότητές που τα χαρακτηρίζει. Επίσης χρησιμοποιείται το tag <text> για σημεία του κειμένου που δεν μπορούν να ταιριάξουν με κάποιο στοιχείο της οντολογίας και τα οποία αργότερα θα μπορούν να εντοπιστούν για επεξεργασία με σκοπό τον εμπλουτισμό της οντολογίας. Ένα παράδειγμα φαίνεται Σχήμα 4.15.
- Τέλος με την διεπαφή που παρέχεται στον χρήστη, του δίνεται η δυνατότητα να δώσει πλέον τα στοιχεία για τα οποία θα ήθελε να πάρει πληροφορία από τα κείμενα. Π.χ. κάποιος που χρειάζεται να πάρει πληροφορία για το σε ποια κύτταρα εκφράζεται το lin-11, θα επέλεγε την κατηγορία “biological process” (με υποκατηγορία “biosynthesis:type=expression”), την κατηγορία “cell or cell group” (με υποκατηγορία “type-name”) και το ακριβές keyword “lin-11”, Έτσι το σύστημα θα αναζητήσει για βιολογικές διαδικασίες τύπου “expression” που γίνονται μεταξύ ενός κυττάρου και του συγκεκριμένου γονιδίου, και λόγω του ότι του βάλαμε την υποκατηγορία “type=name” θα προσπαθήσει να εντοπίσει τα κύτταρα που εμφανίζονται στη βιβλιογραφία με συγκεκριμένα ονόματα και όχι απλά σαν κατηγορία. Επίσης κάποιος χρήστης θα μπορούσε να αναζητήσει πιο γενική πληροφορία, π.χ. ποια γονίδια αλληλεπιδρούν με κάποια άλλα, τότε το σύστημα θα αναζητούσε για κατηγορίες και θα επέστρεφε όλους τους δυνατούς συνδυασμούς που εντοπίζονται στη βιβλιογραφία.



Σχήμα 4.15 Παράδειγμα Επισημείωσης Κειμένου από το Textpresso. [34]

Οι Jan Paralic et. al [35] δίνουν μια προσέγγιση τον τομέα ανάκτησης πληροφορίας. Το σύστημά τους είναι βασισμένο σε ένα σχήμα αναπαράστασης πληροφορίας πεδίου, δηλαδή βασισμένο σε μια οντολογία και ονομάζεται Webocraft. Οι νέες πηγές πληροφορίας που προσαρτούνται στο σύστημα, συνδέονται στις έννοιες της οντολογίας αυτής. Με τον τρόπο αυτό οι πηγές μπορούν να ανακτηθούν βάση συσχετίσεων και όχι μόνο με το μερικό ή ακριβές ταίριασμα όρων που γίνεται σε άλλα μοντέλα, όπως το vector model. Οι συγγραφείς παρουσιάζουν τρία διαφορετικά μοντέλα ανάκτηση πληροφορίας και κάνουν σύγκριση μεταξύ τους. Τα τρία μοντέλα είναι τα εξής:

- **Vector Representation Approach:** Η πολύ γνωστή αυτή προσέγγιση είναι βασισμένη στην αναπαράσταση των εγγράφων μέσω διανυσμάτων. Αρχικά κάθε έγγραφο περνά από μια διαδικασία προεπεξεργασίας μέσω της οποίας ως τελικό προϊόν εξάγεται ένα διάνυσμα με κάποια βάρη για τον κάθε όρο του εγγράφου. Το διάνυσμα αυτό είναι και η αναπαράσταση του εγγράφου. Τα βάρη αυτά υπολογίζονται με το πολύ γνωστό tf-idf σχήμα: $w_{ij} = tf_{ij} \times idf_i$, όπου $tf_{ij} = \frac{freq_{ij}}{\max_e freq_{ej}}$ και $idf_i = \log\left(\frac{N}{n_i}\right)$,

  $freq_{ij}$ είναι ο αριθμός των εμφανίσεων του όρου t_i στο έγγραφο d_j, N είναι το πλήθος της συλλογής των εγγράφων και n_i είναι το πλήθος των εγγράφων στα οποία εμφανίζεται ο όρος t_i.

Κάθε τέτοιο διάνυσμα εισέρχεται σε έναν πίνακα Α που αποτελεί την αναπαράσταση ολόκληρης της συλλογής των εγγράφων.

Για να είναι το σύστημα σε θέση να βρει κάποια έγγραφα σχετικά με κάποιο συγκεκριμένο ερώτημα $\vec{Q}$, είναι απαραίτητο να αναπαρασταθεί το ερώτημα $\vec{Q}$ με τον ίδιο τρόπο όπως και κάθε έγγραφο $\vec{D}_i$ (π.χ. με ένα διάνυσμα των βαρών του κάθε όρου). Η ομοιότητα μεταξύ του ερωτήματος $\vec{Q}$ και ενός εγγράφου $\vec{D}_i$ υπολογίζεται με το cosine similarity μέτρο:

$$sim_{TF-IDF}(\vec{Q},\vec{D}) = \frac{\vec{D}_i \times \vec{Q}}{\left|\vec{D}_i\right|\left|\vec{Q}\right|}$$

- **Latent Semantic Indexing Approach:** Η LSI προσέγγιση είναι βασισμένη στην Singular Value Decomposition του tf-idf πίνακα Α. Με τη μέθοδο αυτή τρεις πίνακες υπολογίζονται :

$$A = USV^T,$$

όπου S ο διαγώνιος πίνακας ιδιοτιμών, και U,V οι πίνακες των αριστερών και δεξιών ιδιοδιανυσμάτων. Αν οι ιδιοτιμές του S ταξινομηθούν κατά φθίνουσα σειρά, οι πρώτες k μεγαλύτερες τιμές μπορούν να διατηρηθούν και οι υπόλοιπες να τεθούν ίσες με 0. Το γινόμενο των πινάκων που προκύπτουν, είναι προσεγγιστικά ίσο με τον Α:

$$A \cong A_{SVD}, \text{ όπου } A_{SVD} = U_k S_k V_k^T$$

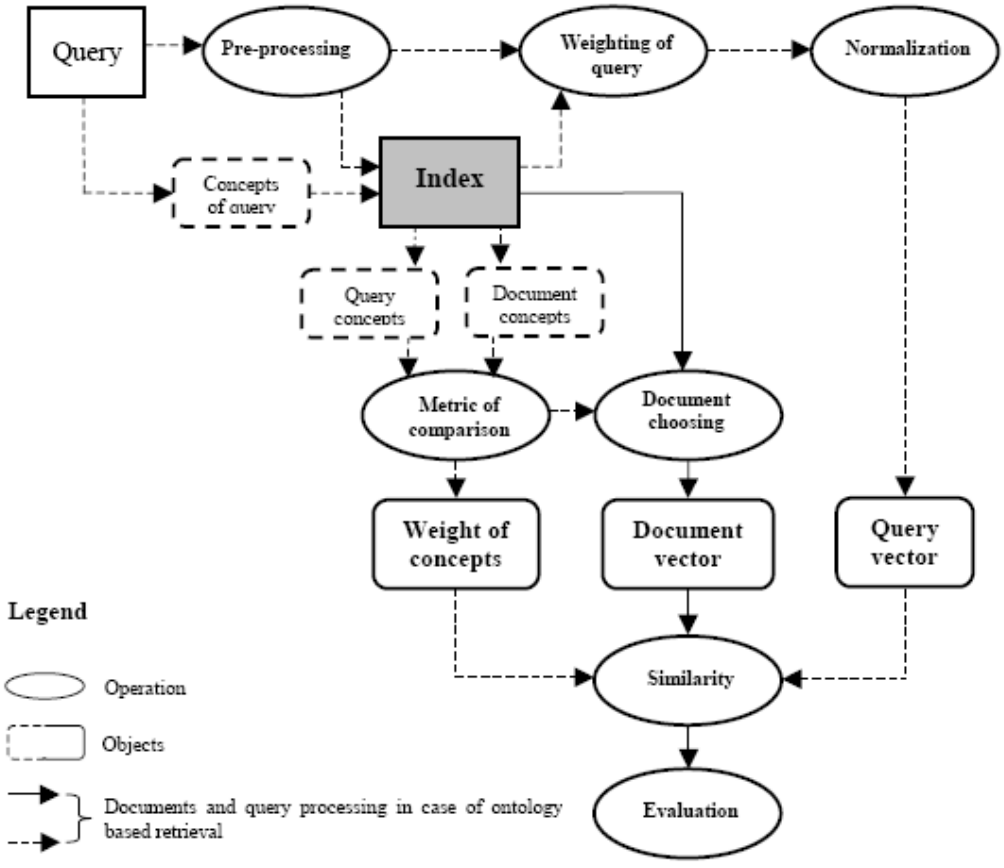
Για να είμαστε σε θέση να ορίσουμε την ομοιότητα μεταξύ ενός ερωτήματος και του προσεγγιστικού διανύσματος ενός εγγράφου $\vec{D}_{i,SVD}$, χρειάζεται ο μετασχηματισμός του διανύσματος του ερωτήματος σε ένα νέο χώρο χαρακτηριστικών.

$$Q_{SVD} = Q_{TF-IDF}^T U_k S_k^{-1}$$

και έπειτα να υπολογιστεί η cosine similarity όπως προηγουμένως:

$$sim_{SVD}(\vec{Q}_{SVD}, \vec{D}_{SVD}) = \frac{\vec{D}_{i,SVD} \times \vec{Q}_{SVD}}{\left|\vec{D}_{i,SVD}\right| \left|\vec{Q}_{SVD}\right|}$$

- **Ontology Based Approach:** Αυτή είναι η προσέγγιση που χρησιμοποιήθηκε στο Webocraft, το σύστημα που ανέπτυξαν οι συγγραφείς για αναζήτηση εγγράφων βασιζόμενη σε μια οντολογία. Στην παρούσα φάση δεν λάβανε υπόψη, τους τύπους των σχέσεων της οντολογίας για τον υπολογισμό της ομοιότητας μεταξύ των concepts. Επιπλέον υποθέσανε πως τα concepts που συσχετίζονται με το ερώτημα ενός χρήστη είναι γνωστά. Ο τρόπος με τον οποίο η προσέγγιση αυτή χειρίζεται ένα ερώτημα φαίνεται στο Σχήμα 4.16.



Σχήμα 4.16 Ο Τρόπος Λειτουργίας του Webocraft. [35]

Για ένα ερώτημα:

- Αρχικά ανακτούνται τα concepts που το χαρακτηρίζουν, στην παρούσα περίπτωση αυτά έχουν δοθεί από τον χρήστη.
- Αμέσως μετά ανακτούνται τα concepts που συσχετίζονται με το κάθε έγγραφο.
- Γίνεται σύγκριση μεταξύ τους με ένα απλό μέτρο, που εκφράζει την ομοιότητα μεταξύ ενός εγγράφου $\vec{D}_i$ και του ερωτήματος $\vec{Q}$.

$$sim_{onto}(\vec{Q},\vec{D}_i)=\begin{Bmatrix}|Q_{con}\cup D_{i,con}|, if\ |Q_{con}\cup D_{i,con}|\neq 0\\ k\end{Bmatrix},$$

όπου Q_{con} το σύνολο των concepts που ορίσθηκαν για το ερώτημα Q, D_{con} το σύνολο των concepts που ορίσθηκαν για το έγγραφο $\vec{D}_i$ και k μια μικρή σταθερά π.χ. 0.1. Το αποτέλεσμα της παραπάνω έκφρασης αποτελεί το ontology based similarity μέτρο. Καλύτερα αποτελέσματα επιτεύχθηκαν συνδυάζοντας το μέτρο αυτό με ένα από τα δυο προηγούμενα, π.χ. το vector model. Ο τελικός βαθμός ομοιότητας υπολογίζεται με έναν πολλαπλασιασμό τους π.χ.:

$$sim(Q,D_i)=sim_{onto}*sim_{TF-IDF}(Q,D_i)$$

Οι A. Hliaoutakis et. al [36] ανέπτυξαν ένα σύστημα για ανάκτηση από ιατρική βιβλιογραφία, το MedSearch. Το σύστημα αυτό υποστηρίζει ανάκτηση με τη βοήθεια της SSRM (Semantic Similarity Retrieval Model) [37], που αποτελεί μια καινοτόμα μέθοδο που είναι ικανή να συσχετίζει έγγραφα που περιέχουν σημασιολογικές ομοιότητες. Η SSRM προτείνει την εύρεση σημασιολογικών ομοιοτήτων μεταξύ όρων σε έγγραφα και ερωτήσεις, χρησιμοποιώντας οντολογίες και συσχετίζοντας τέτοιους όρους χρησιμοποιώντας μεθόδους σημασιολογικής ομοιότητας. Η SSRM υπόσχεται πολύ καλή απόδοση πετυχαίνοντας πολύ καλύτερα precision και recall από την VSM (Vector Space Model) για ανάκτηση από το Medline. Συνήθως στις μεθόδους ανάκτησης πληροφορίας, τα έγγραφα αναπαρίστανται από διανύσματα και κάθε όρος αρχικά αναπαρίσταται από το βάρος tf • idf. Για μικρά ερωτήματα που περιέχουν λίγους όρους τα βάρη αρχικοποιούνται στο 1. Η SSRM λειτουργεί σε τρία βήματα:

- **Term Re-Weighting:** Το βάρος q_i του κάθε όρου του ερωτήματος τροποποιείται ανάλογα με την συσχέτισή του με σημασιολογικά όμοιους όρους j, του ίδιου διανύσματος (του ερωτήματος). $q_i = q_i + \sum_{sim(i,j)\geq t}^{j\neq i} q_j sim(i,j)$ (1),

  όπου t είναι ένα όριο δοσμένο από τον χρήστη (εδώ οι συγγραφείς το θέτουν ίσο με 0.8). Η σημασιολογική ομοιότητα μεταξύ όρων υπολογίζεται σύμφωνα με τη μέθοδο που δίνεται στο [37]. Με την τροποποίηση αυτή των βαρών, συσχετιζόμενοι όροι μέσα στο ερώτημα ενισχύουν ο ένας τον άλλο.
- **Term Expansion:** Αρχικά το ερώτημα εμπλουτίζεται με συνώνυμους όρους και έπειτα με σημασιολογικά όμοιους όρους που βρίσκονται υψηλότερα ή χαμηλότερα στην ταξινομία της οντολογίας. Ερευνάται η γειτονιά του όρου στην ταξινομία και όλοι οι όροι με ομοιότητα μεγαλύτερη από ένα κατώφλι T (οι συγγραφείς εδώ το ορίζουν ως T=0.9), εισέρχονται στο διάνυσμα του ερωτήματος. Σημειώνεται πως αν το T είναι αρκετά μικρό, τότε είναι πιθανόν να εισέλθουν πολλοί νέοι όροι και να διαστρεβλωθεί το θέμα του ερωτήματος. Η διαδικασία αυτή μπορεί να εισάγει όρους περισσότερου του ενός επιπέδου χαμηλότερα ή υψηλότερα από τον αρχικό όρο. Πλέον επαναυπολογίζεται το βάρος του κάθε όρου του διανύσματος του ερωτήματος ως εξής: $q'_i = q_i + \sum_{sim(i,j)\geq T, j\in Q}^{i\neq j} \frac{1}{n} q_j sim(i,j)$ (2),

  όπου n είναι το πλήθος των υπωνύμων (hyponyms) του κάθε όρου j που επεκτάθηκε, q_i είναι το βάρος του όρου i πριν την επέκταση και Q είναι το υποσύνολο του συνόλου των αρχικών όρων του ερωτήματος που οδήγησαν σε νέους όρους που προστέθηκαν στο επεκταμένο ερώτημα. Για τα υπερνύμια (hypernyms) το n=1. Το $q_i = 0$ αν ο όρος i δεν υπήρχε στο αρχικό ερώτημα αλλά, εισήλθε κατά τον εμπλουτισμό.
- **Document Similarity:** Η ομοιότητα μεταξύ ενός επεκταμένου ερωτήματος q και ενός εγγράφου d υπολογίζεται ως εξής: $Sim(q,d) = \frac{\sum_i \sum_j q_i d_j sim(i,j)}{\sum_i \sum_j q_i d_j}$ (3), όπου i και j είναι όροι στο ερώτημα και το έγγραφο αντίστοιχα. Το παραπάνω μέτρο ομοιότητας κανονικοποιείται στην κλίμακα [0,1].

Το MedSearch υποστηρίζει ανάκτηση βιβλιογραφικών πληροφοριών από το Medline χρησιμοποιώντας το VSM μοντέλο καθώς και το SSRM μοντέλο που έχει το MesH σαν αναφορική οντολογία. Το VSM καθώς και το SSRM υλοποιούνται με το Lucene που αποτελεί μια πλήρους εξοπλισμένη βιβλιοθήκη αναζήτησης κειμένων σε Java. Όλα τα έγγραφα περιέχουν τίτλο, abstract και MeSH όρους. Αυτές οι περιγραφές αναλύονται συντακτικά και δημιουργούνται ξεχωριστά διανύσματα από MeSH όρους, για τα οποία και θα γίνει υπολογισμός της σημασιολογικής τους ομοιότητας με τα ερωτήματα, σύμφωνα με την εξίσωση 3. Τα βάρη όλων των MeSH όρων αρχικοποιούνται στο 1, ενώ τα βάρη των τίτλων και των abstracts αρχικοποιούνται με την τιμή tf•idf. Τελικά η ομοιότητα μεταξύ ενός ερωτήματος και ενός εγγράφου υπολογίζεται ως εξής:

$$Sim(q,d) = Sim(q,d_{MeSH-terms}) + Sim(q,d_{title}) + Sim(q,d_{abstract}) \quad (4),$$

όπου $d_{MeSH-terms}$, d_{title}, $d_{abstract}$ είναι οι αναπαραστάσεις των MeSH όρων, του τίτλου και του abstract του εγγράφου αντίστοιχα. Ο τύπος αυτός, προτείνει πως ένα έγγραφο είναι πιο κοντά σε ένα ερώτημα, όταν τα περισσότερα συστατικά του είναι κοντά στο ερώτημα αυτό. Τα αποτελέσματα της αξιολόγησης που πραγματοποιήσαν, δείξαν πως η SSRM μέθοδος είναι πιο αποτελεσματική από την κλασική VSM μέθοδο, και πετυχαίνει 20% καλύτερο precision και 20% καλύτερο recall. Στα πειράματα που κάνανε αποδείχθηκε πως με τις συγκεκριμένες παραδοχές η τεχνική ανάκτησης που προτείναν, βασισμένη σε οντολογία, είναι πολλά υποσχόμενη, παρέχοντας καλύτερη απόδοση ανάκτησης από άλλες τεχνικές.

4.4. Ιατρικές Οντολογίες με Χρήση Natural Language Processing Tools (NLP)

Η δημιουργία οντολογιών from scratch είναι μια δύσκολη και επίπονη διαδικασία. Επίσης το μεγαλύτερο μέρος της πληροφορίας βρίσκεται σε γραπτά κείμενα και αναφορές. Έτσι πολλοί ερευνητές προσανατολίζονται στη δημιουργία οντολογιών είτε με τη χρήση ήδη έτοιμων μοντέλων αναπαράστασης γνώσης, είτε με την αυτόματη ή ημιαυτόματη δημιουργία οντολογιών, εξάγοντας τα στοιχεία τους μέσα από γραπτά κείμενα και αναφορές του πεδίου ενδιαφέροντος, με τη χρήση εργαλείων επεξεργασίας φυσικής γλώσσας (NLP).

Οι Saurav Sahay et al. [83] στην εργασία τους «Domain Ontology Construction from Biomedical Text», πραγματεύονται τη δημιουργία οντολογιών πεδίου, και πιο συγκεκριμένα ιατρικών πεδίων, με τη χρήση εργαλείων επεξεργασίας φυσική γλώσσας σε κείμενα του

πεδίου. Οι συγγραφείς αναφέρουν την ύπαρξη οντολογιών που αφορούν την ιατρική, όπως είναι το UMLS, αλλά επισημαίνουν το πρόβλημα πως τέτοιες οντολογίες δεν μπορούν να αποδώσουν με μεγάλη λεπτομέρεια την αναλυτική περιγραφή εξειδικευμένων πεδίων της ιατρικής και σαν ένα παράδειγμα δίνουν πως μπορεί το UMLS να συμπεριλαμβάνει πολλές ασθένειες, ιούς και βακτήρια, αλλά δεν περιλαμβάνει πληροφορίες συσχέτισης μεταξύ ασθενειών και των αιτιών που τις προκαλούν. Στην εργασία τους που έχει σαν πεδίο εφαρμογής την πυρηνική καρδιολογία (Nuclear Cardiology), συλλέγουν ένα σύνολο από abstracts, από δημοσιεύσεις που αφορούν το πεδίο, και μέσω αυτών προσπαθούν να εξάγουν έννοιες του πεδίου και σχέσεις μεταξύ τους. Με τον τρόπο αυτό θα μπορέσουν να δημιουργήσουν μια οντολογία του πεδίου ενδιαφέροντος, με πιο εξειδικευμένη πληροφορία, σε σχέση με αυτή που παρέχει το UMLS. Οι συγγραφείς κάνουν χρήση κάποιον εργαλείων επεξεργασίας φυσικής γλώσσας ώστε να πάρουν φράσεις που είναι σχετικές με το πεδίο ενδιαφέροντος από τα abstracts των δημοσιεύσεων, να τους αναθέσουν κάποια concepts σχετικά με το UMLS, ώστε να τις κατατάξουν σε κάποια σημασιολογική κατηγορία, και να βρουν σχέσεις μεταξύ τους, ώστε να μπορέσουν να τις συσχετίσουν, και τελικά να δημιουργηθεί μια οντολογία του πεδίου. Κάνουν χρήση του MMTx [73] μέσω του οποίου βρίσκουν φράσεις του πεδίου συσχετισμένες με υποψήφια concepts από το UMLS. Επίσης, βρίσκουν αλληλουχίες τύπου Υποκείμενο – Ρήμα – Αντικείμενο, μέσω των οποίων βρίσκουν συσχετίσεις των εννοιών που εντόπισαν στο προηγούμενο βήμα, και με τον τρόπο αυτό δημιουργούν ένα σημασιολογικό δίκτυο. Τέλος, μέσω κάποιων σημασιολογικών και λεξικογραφικών συσχετισμών κάνουν προσπάθεια να αντιστοιχίσουν το πιο κοντινό concept από τα υποψήφια που βρέθηκαν στην κάθε έννοια – φράση του πεδίου ενδιαφέροντος, όπως επίσης μέσω patterns προσπαθούν να αποδώσουν στα ρήματα που αποτελούν τις συσχετίσεις των εννοιών του πεδίου, κάποια σημασιολογική κατηγορία από αυτές που εμπεριέχονται στο σημασιολογικό δίκτυο του UMLS.

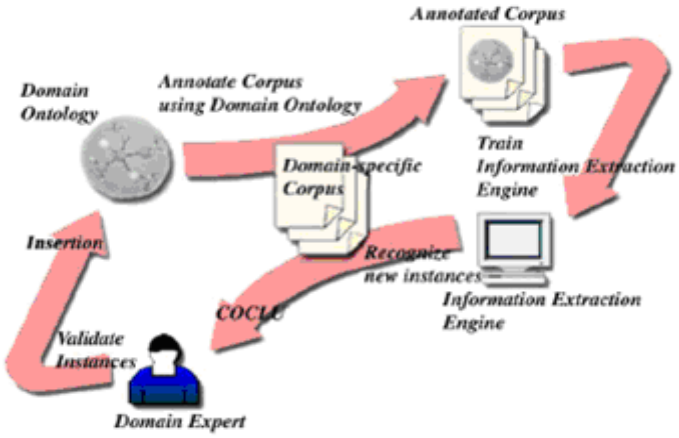
Οι A. G. Valarakos et al. στη δημοσίευσή τους «Building an allergens ontology and maintaining it using machine learning techniques» [38], παρουσιάζουν και αυτοί με τη σειρά τους τη μεγάλη σημασία των οντολογιών στη διευκόλυνση του διαμοιρασμού και επαναχρησιμοποίησης της γνώσης μεταξύ ανθρώπων αλλά και συστημάτων. Στην εργασία τους παρουσιάζουν μια μεθοδολογία για τη δημιουργία μιας καλά ορισμένης οντολογίας, διατηρώντας την, εκμεταλλευόμενοι τεχνικών μηχανικής μάθησης (machine learning techniques). Εφαρμόζουν τη μεθοδολογία αυτή στην περιοχή των αλλεργιογόνων (allergens)

και παρουσιάζουν τον τρόπο σκέψης τους για το χτίσιμο της οντολογίας, τις τεχνικές που χρησιμοποίησαν καθώς και την αξιολόγησή της. Η οντολογία που δημιουργούν όπως προαναφέραμε έχει ως επίκεντρό το πεδίο των αλλεργιογόνων. Οι συγγραφείς αναφέρουν πως το πεδίο της βιοϊατρικής είναι ένα πεδίο το οποίο βρίσκεται συνεχώς σε εξέλιξη. Τα διάφορα περιοδικά τα οποία συγκεντρώνουν τις δημοσιεύσεις στο πεδίο αυτό, εμπλουτίζονται συνεχώς με νέα πληροφορία, όπως επίσης και οι διάφορες βάσεις δεδομένων που περιέχουν πληροφορία για το πεδίο και που βρίσκονται διάσπαρτες ανά τον κόσμο. Έτσι τονίζουν πως η ανάγκη χρήσης τεχνικών (intelligent techniques) για αυτόματη εξαγωγή πληροφορίας (knowledge retrieval) είναι επιτακτική ανάγκη για τόσο ραγδαία αναπτυσσόμενα πεδία όπως την βιοϊατρική. Επίσης αναφέρουν πως η χρήση οντολογιών που περιγράφουν και μοντελοποιούν την ορολογία και την πληροφορία ενός πεδίου είναι το βασικό στοιχείο για τέτοιες τεχνικές. Γενικά οι συγκεκριμένοι συγγραφείς παρουσιάζουν μια μεθοδολογία για την υλοποίηση και διατήρηση μιας οντολογίας. Επίσης ακολουθούν τη μεθοδολογία αυτή για την υλοποίηση μιας οντολογίας και εφαρμογή της, στο πεδίο των αλλεργιογόνων και αξιολογούν την όλη διαδικασία. Τα στάδια της ανάπτυξης μιας οντολογίας τα οποία είναι ευρέως αποδεκτά είναι: το specification (όπου ορίζεται ο σκοπός δημιουργίας της οντολογίας, ώστε να περιοριστούν τα εννοιολογικά μοντέλα που μπορούν να χρησιμοποιηθούν για την αναπαράσταση ενός πεδίου ενδιαφέροντος), το conceptualization (όπου απαριθμούνται όροι που αναπαριστούν τα concepts, τις ιδιότητές τους και τις σχέσεις που υπάρχουν μεταξύ τους, με σκοπό να αποτελέσουν την εννοιολογική περιγραφή της οντολογίας), το formalization (όπου η εννοιολογική περιγραφή του προηγούμενου σταδίου μετασχηματίζεται σε ένα formal μοντέλο. Αυτό γίνεται δίνοντας τους ορισμούς των concepts μέσω αξιωμάτων που περιορίζουν την πιθανή παρερμηνεία των εννοιών που θέλουν να αποδώσουν, όπως επίσης μέσω των σχέσεων που οργανώνουν τα concepts π.χ. is-a και part-of σχέσεων), το implementation (εδώ η οντολογία που παράχθηκε στο προηγούμενο στάδιο, υλοποιείται χρησιμοποιώντας μια γλώσσα αναπαράστασης γνώσης – knowledge representation language) και το maintenance (όπου η υλοποιημένη οντολογία ενημερώνεται και ελέγχεται η ορθότητά της). Το τελευταίο στάδιο είναι πολύ σημαντικό αφού οι οντολογίες χρησιμοποιούνται σε πρακτικές εφαρμογές, οπότε και πρέπει να λαμβάνουν υπ' όψιν τους αλλαγές που υφίστανται στη γνώση που αναπαριστούν π.χ. ενσωμάτωση νέων instances ή ποικιλομορφίες των ήδη υπαρχόντων, ενσωμάτωση νέων concepts κ.α. Οι συγγραφείς παρουσιάζουν αρχικά τα κριτήρια σχεδίασης που ακολούθησαν στην περίπτωσή τους, για τα τέσσερα πρώτα στάδια ανάπτυξης. Στη διάρκεια των τεσσάρων αυτών σταδίων ακολούθησαν τα βασικά και ευρέως

αποδεκτά κριτήρια σχεδίασης οντολογιών: Σαφήνεια (coherence) που δηλώνει την απαίτηση να υπάρχουν ορισμοί που να είναι ακριβής και ξεκάθαροι. Το κριτήριο αυτό αναφέρεται κυρίως στο δεύτερο στάδιο ανάπτυξης (conceptualization) όπου περιγράφεται το εννοιολογικό μοντέλο της οντολογίας (concepts, attributes, relations). Επίσης συσχετίζεται και με το τρίτο στάδιο (formalization) αφού η formal μοντελοποίηση (formal models) παρέχει το μέσο για τον ορισμό των concepts με αναγκαίες και ικανές συνθήκες. Συνοχή (coherence), που δηλώνει την ανάγκη οι ορισμοί να είναι συνεπής. Σε μια καλά ορισμένη οντολογία πρέπει να υπάρχει λογική συνοχή μεταξύ των στοιχείων της. Επεκτασιμότητα (extendibility), που δηλώνει πως η οντολογία πρέπει να επιτρέπει την ενσωμάτωση νέων concepts χωρίς την ανάγκη να αναθεωρηθεί η μέχρι τώρα δομή της. Ελάχιστη εξειδίκευση κωδικοποίησης (minimal encoding bias), που δηλώνει την ανάγκη να μην χρησιμοποιούνται συγκεκριμένες αναπαραστάσεις της γνώσεις για χάρη της υλοποίησης. Έτσι η οντολογία θα είναι ανεξάρτητη από την εφαρμογή στην οποία θα χρησιμοποιηθεί. Ελάχιστη οντολογική αφοσίωση (Minima ontological commitment), που δηλώνει την ανάγκη να ορισθούν τόσα concepts, attributes και relations όσα είναι απαραίτητα για την αναπαράσταση της γνώσης στο συγκεκριμένο πεδίο. Όσο λιγότερο αφοσιωμένη είναι μια οντολογία, τόσο περισσότερο μπορεί να επαναχρησιμοποιηθεί σε νέες εφαρμογές. Επίσης οι συγγραφείς αναφέρουν πως, ναι μεν πρέπει να ακολουθούνται τα παραπάνω κριτήρια, αλλά στην πράξη, στις περισσότερες εφαρμογές χρειάζεται να γίνονται μερικές εναλλαγές, όπως πχ. στην περίπτωση της ελάχιστης οντολογικής αφοσίωσης, αν η οντολογία παρέχει λίγη πληροφορία για ένα πεδίο τότε επηρεάζει την εμβέλεια των εφαρμογών που μπορούν πραγματικά να τη χρησιμοποιήσουν.

Μια από τις βασικές ανησυχίες τους στη διάρκεια του conceptualization, ήταν η αποφυγή ύπαρξης διπλότυπων μονάδων γνώσης (knowledge units). Όσες περισσότερε φορές εμφανίζεται μια μονάδα γνώσης σε διαφορετικά σημεία μέσα στην οντολογία, τόσο μεγαλώνει η ασάφειά της, πράγμα που επηρεάζει την σαφήνεια (clarity) της οντολογίας. Επίσης, για χάρη της σαφήνειας, απορρίπταν μονάδες γνώσης που δεν ήταν άμεσα χρήσιμες στη μοντελοποίηση του πεδίου, και σε άλλες περιπτώσεις κατηγοριοποιούσαν περαιτέρω τα στιγμιότυπα ενός concept εισάγοντας επιπλέον concepts σαν παιδιά του. Επιπλέον δημιουργούσαν μικρές οντολογίες (subontologies) όπου θεωρούσαν πως κάποιες μονάδες γνώσης μπορούσαν να συγκροτήσουν αυτόνομες οντολογίες που αργότερα θα συνεργαζόταν με άλλες. Οι μικρότερες οντολογίες (subontologies) μπορούν να διατηρούνται ευκολότερα

από ότι μια ολοκληρωμένη οντολογία ενός πεδίου ενδιαφέροντος. Επίσης μπορούν να χρησιμοποιηθούν από άλλες οντολογίες σε συναφείς εφαρμογές. Με αυτόν τον τρόπο χειρίστηκαν την Linnaeus ταξινομία (Linnaeus taxonomy) στην οντολογία για το πεδίο των αλλεργιογόνων. Για το στάδιο του formalization βασίστηκαν σε προηγούμενη δουλειά [39] για τη δημιουργία αυστηρών ορισμών. Όσον αφορά το στάδιο υλοποίησης (implementation stage) χρησιμοποίησαν την ευρέως διαδεδομένη γλώσσα υλοποίησης οντολογιών, την OWL. Στο τελευταίο στάδιο ανάπτυξης της οντολογίας (maintenance stage), το οποίο περιλαμβάνει την ενσωμάτωση στιγμιότυπων και ενέργειες εμπλουτισμού της οντολογίας, οι συγγραφείς δίνουν μια μεθοδολογία για αυτοματοποίησή του. Γενικά η διαδικασία που προτείνεται είναι η εξής: η ύπαρξη μιας αρχικής οντολογίας (initial ontology) η οποία θα περιέχει ένα μικρό αριθμό μονάδων πληροφορίας όπως και ένα σύνολο εγγράφων (corpus) για το συγκεκριμένο πεδίο ενδιαφέροντος. Αυτή η αρχική οντολογία χρησιμοποιείται για αυτόματη επισημείωση (annotate) του corpus (ontology-based semantic annotation). Το επισημειωμένο corpus (annotated corpus) χρησιμοποιείται για την εκπαίδευση μιας μηχανής εξαγωγής πληροφορίας (information extraction engine), η οποία έπειτα ανακαλύπτει νέα στιγμιότυπα (knowledge discovery). Αμέσως μετά γίνεται αναζήτηση για τυπογραφικές ποικιλομορφίες των υπαρχόντων στιγμιότυπων (knowledge refinement). Τα αυτόματα αποκτηθέντα στιγμιότυπα και οι τυπογραφικές ποικιλομορφίες παρουσιάζονται σε έναν ειδικό του πεδίου, ο οποίος τα εξετάζει και αποφασίζει αν πρέπει να ενσωματωθούν στην οντολογία (knowledge validation). Αυτή η διαδικασία γίνεται επαναληπτικά, όπου η νέα έκδοση της οντολογίας χρησιμοποιείται σε κάθε επανάληψη. Η όλη διαδικασία μπορεί να απεικονιστεί στο Σχήμα 4.17.



Σχήμα 4.17 Διαδικασία Αυτόματου Εμπλουτισμού μιας Οντολογίας. [38]

Γενικά στην ontology-based semantic annotation χρησιμοποιούνται διάφορες string matching τεχνικές. Στη συνέχεια στο knowledge discovery χρησιμοποιούν μια machine learning-based information extraction engine που εκπαιδεύεται από την πληροφορία που εξήχθη από το προηγούμενο στάδιο. Εδώ μπορούν να χρησιμοποιηθούν διάφορες τεχνικές για information extraction, και σε αντίθεση με άλλες προσεγγίσεις η δική τους εκμεταλλεύεται ένα αυτόματα επισημειωμένο corpus για να μειωθεί η ανθρώπινη παρέμβαση. Στο knowledge refinement στάδιο εισάγουν μια νέα relation την "hasTypographicVariants" που δείχνει να είναι πολλά υποσχόμενη στον τομέα της βιοϊατρικής όπου τα ονόματα πολλών οντοτήτων δεν είναι γραμμένα σε μια κοινώς αποδεκτή ορολογία. Στο στάδιο αυτό χρησιμοποιούν έναν compression-based clustering αλγόριθμο, τον COLCU. Τέλος, στο knowledge validation στάδιο οι ειδικοί του πεδίου αξιολογούν τα instances, attribute values και typographic variants που εξήχθησαν και αποφασίζουν την ενσωμάτωσή τους ή μη στην οντολογία.
Επίσης, δίνεται μια περιγραφή της αξιολόγηση της όλης διαδικασίας. Το πρώτο στάδιο για την αξιολόγηση αυτή είναι η προσεκτική συλλογή ενός καλού συνόλου από έγγραφα (corpus). Επίσης, πρέπει να δημιουργηθεί μια gold ontology με την οποία πρέπει να συγκρίνονται οι διάφορες εκδόσεις των εξαγόμενων οντολογιών του κάθε σταδίου. Τέλος, θα πρέπει να ορισθούν οι διάφοροι παράμετροι των πειραμάτων, όπως επίσης και το μέγεθος της αρχικής οντολογίας.

Η Εργασία των Baneyx et al. [41], πραγματεύεται την ανάπτυξη μιας μεθοδολογίας για την δημιουργία οντολογιών μέσω γραπτών κειμένων, στην οποία το μεγαλύτερο μέρος της δουλείας θα την αναλαμβάνει ένας knowledge Engineering και όχι ο γιατρός που συσχετίζεται με το πεδίο του ενδιαφέροντος. Για περίπου 10 χρόνια τα δημόσια νοσοκομεία της Γαλλίας, είχαν την ανάγκη να ανταλλάσουν πληροφορίες για τις δραστηριότητές τους. Για κάθε ασθενή, οι πληροφορίες συλλέγονται στο εξιτήριό του, χρησιμοποιώντας τη διεθνή ταξινόμηση των ασθενειών "CIM-10" για την κωδικοποίηση των διαγνώσεων, και την "CCAM" για την κωδικοποίηση των δραστηριοτήτων τους. Η Γαλλική διαδικασία κωδικοποίησης ως τώρα , γινόταν συνήθως χειροκίνητα από τους γιατρούς, χρησιμοποιώντας έναν θησαυρό "thesauri" του συγκεκριμένου πεδίου. Όμως, έχει σημειωθεί στη βιβλιογραφία πως η αυτοματοποίηση της διαδικασίας κωδικοποίησης χρειάζεται μια εννοιολογική οργάνωση των ιατρικών όρων, που το νόημά τους θα είναι ενσωματωμένο μέσα στην ίδια τη δομή του μοντέλου [42], δηλαδή χρειάζεται μια οντολογία. Η δουλειά τους ήταν μέρος του PERTOMED Project, που είχε ως αντικείμενο, την ανάπτυξη μιας πλατφόρμας διαδικτύου,

που θα παρείχε μεθόδους και εργαλεία για την παραγωγή και χρήση οντολογικών πηγών στο ιατρικό πεδίο. Στη συγκεκριμένη εργασία στόχος ήταν να παρασχεθεί βοήθεια στους πνευμονολόγους στην κωδικοποίηση των δραστηριοτήτων και διαγνώσεών τους, με λογισμικό. Το λογισμικό αυτό θα αναπαριστά την ιατρική γνώση με μια οντολογία της συγκεκριμένης ειδικότητας της ιατρικής.

Όπως προαναφέρθηκε βασικός στόχος των ερευνητών, είναι η οντολογία να δημιουργηθεί επί το πλείστον από έναν knowledge engineer παρά από έναν γιατρό. Η βασική δυσκολία σε μια τέτοια διαδικασία είναι η αναγνώριση και ταξινόμηση των εννοιών (concepts) του δοθέντος πεδίου. Για το λόγο αυτό χρησιμοποίησαν γραπτές αναφορές σαν τη βασική πηγή πληροφοριών, και διαδικασίες ανάλυσης φυσικής γλώσσας (NLP) για την ανάλυσή τους. Επίσης, λόγω του ότι βασίζονται στην υπόθεση, πως ο πιο φυσικός τρόπος να εκφραστούν επακριβώς τα concepts της οντολογίας είναι η επεξήγησή τους σε φυσική γλώσσα [42], η μέθοδος που χρησιμοποίησαν έχει ως βάση τις differential semantics principles [43]. Η βασική διαφοροποίηση των differential ontologies, σε σχέση με τις υπόλοιπες οντολογίες, είναι πως δίνουν πολύ μεγάλη έμφαση στην επακριβή θέση ενός concept στο οντολογικό δέντρο. Σε αυτές τις οντολογίες, η σημασία ενός concept, δίνεται συλλέγοντας όλες τις ομοιότητες και διαφορές που εκφράζονται σε φυσική γλώσσα, μεταξύ του κόμβου-στόχου και της ρίζας του δέντρου. Με άλλα λόγια, η σημασία ενός concept, δίνεται από τη θέση του στην ιεραρχία της οντολογίας. Η μέθοδός τους έχει ως βασική υπόθεση το συντονισμό δύο μεθόδων για το χτίσιμο της οντολογίας: i) μια μέθοδος που αποτελείται από τη δημιουργία πηγών ορολογίας με distributional analysis [44] και ii) μια μέθοδο βασισμένη στην εύρεση σημασιολογικών σχέσεων, από την παρατήρηση ακολουθιών σε ένα corpus [45].

Για να καλύψουν όσο διεξοδικά γίνεται την όλη περιοχή των πνευμονολογικών ασθενειών, συλλέξανε 1038 εξιτήρια ασθενών, από έξι νοσοκομεία. Σε μια προηγούμενη δουλειά είχε αποδειχθεί πως 350.000 λέξεις είναι ένας καλός αριθμός για την εξαγωγή καλών αποτελεσμάτων [46]. Αυτό το 1^{o} corpus [PDS], έχει περίπου 417.000 λέξεις, που σημαίνει πως είναι μια καλή βάση για το πείραμα. Πρόσθεσαν επίσης ένα βιβλίο διδασκαλίας (corpus με όνομα [BOOK]), που θα βοηθούσε στην βελτίωση και τον έλεγχο της ιεραρχίας της οντολογίας κατά τη φάση της ανάπτυξής της.

Σαν NLP εργαλείο, χρησιμοποίησαν το SYNTEX-UPERY. Ο SYNTEX είναι ένας συντακτικός αναλυτής, βασισμένος στην υπόθεση όμοιων εξαρτήσεων μεταξύ όρων που έχουν κοντινό νόημα. Έτσι ο αναλυτής αυτός επιτρέπει την εύρεση σχέσεων συντακτικών εξαρτήσεων μεταξύ όρων. Στο τέλος της επεξεργασίας, μας δίνει ένα δίκτυο συντακτικών εξαρτήσεων, των οποίων τα μέλη είναι οι όροι που θα χρησιμοποιηθούν για τη δημιουργία της οντολογίας. Έπειτα ο UPERY συνεχίζει σε μια κατανεμημένη ανάλυση (distributional analysis): υπολογίζει κατανεμημένες συγγένειες (distributional proximities) μεταξύ όρων στα κείμενα, με βάση κοινά συντακτικά συμφραζόμενα, και αξιοποιεί όλο το προηγούμενο δίκτυο για να ομαδοποιήσει τους όρους που χρειάζονται. Το υλικό τους προς εκμετάλλευση αποτελείται τελικά από: το δίκτυο των όρων, τις συσχετίσεις συμφραζόμενων και τους συνδέσμους με τα κείμενα. Ο DOE (differential ontology editor) επιτρέπει το χτίσιμο της οντολογίας, λαμβάνοντας υπόψη τις differential semantic principles [43], αλλά δεν είναι δυνατόν να κάνουν formalize την οντολογία με το εργαλείο αυτό, οπότε μετά η οντολογία, εξάγεται σε OWL. Αυτό το format επιτρέπει την διαλειτουργικότητα του μοντέλου και κάνει την οντολογία διαθέσιμη για το Protégé. Το Protégé επιτρέπει να γίνει formalize της οντολογίας με περιγραφική λογική (description logic) και τον ορισμό σχέσεων μεταξύ των concepts.

Οι συγγραφείς ξεχωρίζουν 5 διαδοχικά βήματα:

1) Τη δημιουργία του corpus και την ανάλυσή του με NLP εργαλεία. Στο βήμα αυτό εφαρμόζουν την distributional analysis στο [PDS] corpus για την εύρεση ενός συνόλου όρων.
2) Την επιλογή των υποψήφιων όρων, που αντιπροσωπεύουν τη γνώση στο πεδίο.
3) Τη σημασιολογική κανονικοποίηση του συνόλου των όρων, με εφαρμογή των differential principles. Στο βήμα αυτό, επεξεργάζονται τις σημασιολογικές σχέσεις που αναγνωρίστηκαν με την παρατήρηση αλληλουχιών στο [BOOK] corpus. Έτσι παίρνουν μια ιεραρχική δομή από concepts και relationships.
4) Ακολουθεί το formalizing της οντολογίας, όπου γίνεται ορισμός των concepts με description logic, περιορισμός των relationships, πρόσθεση αξιωμάτων και στιγμιότυπων κ.α.
5) Η μεταφορά της οντολογίας σε μια γλώσσα κατανοητή από υπολογιστή.

Τα [PDS] και [BOOK] corpus έχουν συλλεχθεί σε ένα format που τα NLP εργαλεία δεν μπορούν να τα επεξεργαστούν. Έτσι μετατρέπονται σε text format, γίνονται ανώνυμα, χωρίζονται σε μικρά τμήματα (segmented) και μπαίνουν ετικέτες για κάθε παράγραφο και πρόταση. Στην τελική τους μορφή έχουμε τα [PDS] και [BOOK] corpus σε XML format. Στη συνέχεια το [PDS] corpus αναλύεται από τον SYNTEX-UPERY. Τα αποτελέσματα της ανάλυσης των υποψήφιων όρων στο [PDS] corpus, επιτρέπουν τη δημιουργία της οντολογίας με τα βασικά της στοιχεία. Ένας υποψήφιος όρος, είναι μια noun phrase, Που αποτελείται από την κεφαλή (head) και την επέκταση (expansion). Π.χ. στην noun phrase "Opacity in the left lung", ο όρος "Opacity" είναι η κεφαλή και ο "in the left lung" είναι η επέκταση. Τα αποτελέσματα από την ανάλυση του [BOOK] corpus, επεξεργάζονται για τον προσδιορισμό "synonymy" και "hyperonymy" σχέσεων μεταξύ των υποψήφιων όρων, χρησιμοποιώντας λεξικο-συντακτικά πρότυπα (lexico-syntactic patterns). Οι σχέσεις αυτές βοηθούν στη δομή της ιεραρχίας των πρωταρχικών (primitive) concepts.

Οι υποψήφιοι όροι, που είναι αντιπροσωπευτικοί για τις πνευμονολογικές ασθένειες, επιλέγονται μεταξύ των αποτελεσμάτων που παρέχονται από τον SYNTEX-UPERY μετά την επεξεργασία του [PDS] σε δύο βήματα:

1) Διατρέχουν όλα τα αποτελέσματα που παρήχθησαν με τη συντακτική ανάλυση και επιλέγονται πρώτα να μελετηθούν οι noun phrases που εμφανίζονται στο corpus περισσότερο από 12 φορές (2% του corpus). Στη συνέχεια εντοπίζουν τους σημαντικότερους άξονες που είναι συγκεκριμένοι (τυπικοί) για το corpus και το ιατρικό πεδίο. Σε κάθε υποψήφιο όρο δίνουν ένα κριτήριο εγκυρότητας (validity criterion) που αντιστοιχεί σε αυτούς τους άξονες, από 1 ως 6: 1 (irrelevant term, axis:other), 2 (όροι που έχουν συμπεριληφθεί στην οντολογία), 3 (axis: symptoms), 4 (axis: pathologies), 5 (axis: treatments) και 6 (axis: examinations). Αρχικά όλοι οι όροι θα έχουν το κριτήριο εγκυρότητας 1, ενώ στο τέλος της διαδικασίας θα πρέπει να έχουν ταξινομηθεί όλοι στο 2. Η επιλογή μέσω του κριτηρίου εγκυρότητας, αφήνει το 35% των υποψήφιων όρων, με τους οποίους μπορεί να δουλευτεί η καρδιά της οντολογίας.

2) Η distributional analysis, συνενώνει όρους που έχουν κοινά συμφραζόμενα (descendants in head and descendants in expansion). Επίσης συνενώνει τα

συμφραζόμενα σε σχέση με τους όρους που μοιράζονται (neighbours in head and neighbours in expansion). Όταν κάποιοι όροι είναι descendants in head, μας δίνεται πληροφορία για το τι θα μπορούσε να είναι child concept, ενώ οι descendants in expansion παρέχουν πληροφορία για τη θέση των concepts στην ιεραρχία. Οι neighbours in head and in expansion επιτρέπουν τη δημιουργία συνόλων από υποψήφιους όρους που είναι σημασιολογικά κοντά με τον όρο που είναι υπό μελέτη. Αυτό το τελευταίο είναι αρκετά χρήσιμο για τον σχηματισμό της ιεραρχικής δομής της οντολογίας, και για το κάθετο και τον οριζόντιο άξονα. Για παράδειγμα θα μπορούσε να γίνει μια σύνδεση του συνόλου Α {effusion, lesion, infection, uncompensation} με το {symptoms}. Στο παράδειγμα αυτό θα μπορούσε να γίνει μια αρχική υπόθεση που θα θεωρούσε πως οι υποψήφιοι όροι του συνόλου Α θα είναι siblings concepts (αφού μοιράζονται κοινή σημασιολογία) και το {symptoms} σαν parent concept του συνόλου αυτού.

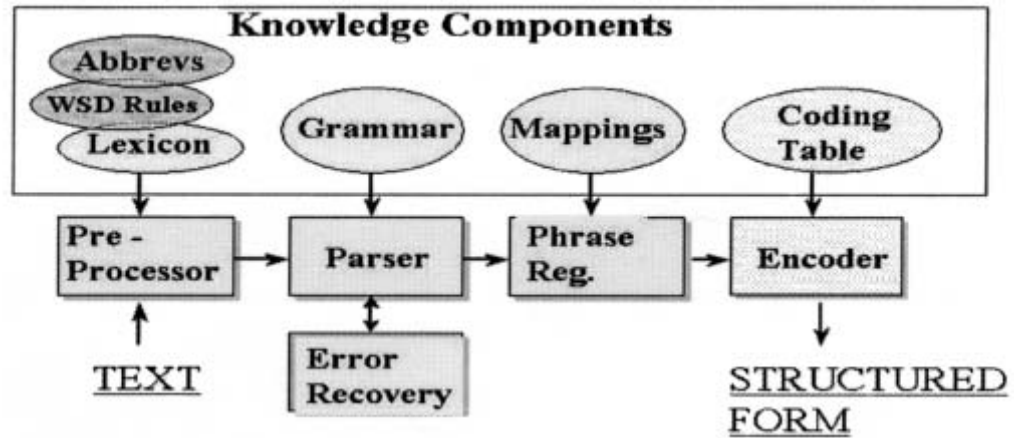
Με σκοπό να αρχίσει να δουλεύεται η ιεραρχία, οι υποψήφιοι όροι οργανώνονται χρησιμοποιώντας τις differential principles που τους ορίζουν. Πρέπει να εκφραστούν σε φυσική γλώσσα οι ομοιότητες και διαφορές κάθε concept με το parent concept του, καθώς και με τα siblings concepts του. Η σημασία ενός concept, όπως προαναφέρθηκε, δίνεται συλλέγοντας όλες τις ομοιότητες και διαφορές, που ορίζονται για κάθε concept μεταξύ του κόμβου στόχου και της ρίζας. Οι τέσσερις άξονες που αναφέρθηκαν νωρίτερα, βελτιώνονται με αυτό τον τρόπο. Επιπρόσθετα χρησιμοποιούνται τα αποτελέσματα της επεξεργασίας του [BOOK] corpus για να βοηθηθεί η εφαρμογή των differential principles. Η ανάλυση βασιζόμενη σε lexico-syntactic patterns recognition, δίνει ενδείξεις στο πως θα εφαρμοστούν οι differential principles. Τα lexico-syntactic patterns είναι κατάλληλα για την εύρεση συγκεκριμένων σημασιολογικών σχέσεων [45]. Τα patterns αυτά έχουν ως βάση έναν marker, που δεικτοδοτεί τη σχέση που θέλουμε, π.χ. kind of για hyperonymy relationships. Για παράδειγμα, ένα pattern του τύπου (NP, kind of, NP) επιτρέπει την εύρεση του hyperonymy συνδέσμου, “Meningitis, kind of pathology” που υποδεικνύει σχέση υπερωνυμίας μεταξύ των “meningitis” και “pathology”. Στη δουλειά αυτή χρησιμοποιήθηκαν 35 markers και 75 patterns. Για τη δημιουργία των differential ontologies, εφάρμοσαν αυτή τη μέθοδο για την εύρεση ορισμών μέσα στο corpus (π.χ. Dry cough is a symptom for bronchitis and it is also a pathology). Τα patterns που χρησιμοποιήθηκαν αναπτύχθηκαν από τους Malaise et al. [47]. Οι όροι που εξήχθησαν ελέγχθηκαν μη

αυτόματα. Στο τέλος αυτών των βημάτων έχουν πλέον μια σημασιολογική κανονικοποίηση του συνόλου των όρων του πεδίου, και έχουν αναπαραστήσει την ιεραρχία των πρωταρχικών concepts και σχέσεων στο DOE.

Όλη αυτή η διαδικασία είναι επαναληπτική, και εφαρμόζεται και για τους υποψήφιους όρους που εμφανίζονται λιγότερο από 12 φορές στο [PDS] corpus. Έτσι εμπλουτίζεται η οντολογία. Τέλος η ιεραρχία που έχει εξαχθεί πρέπει να ελεγχθεί από πνευμονολόγο.

Οι C. Friedman et. al στο [48] περιγράφουν μια μέθοδο κωδικοποίησης που χρησιμοποιεί NLP τεχνικές για την παραγωγή δομημένης, κωδικοποιημένης εξόδου. Η Έξοδος αυτή, αποτελείται από λέξεις σχετικές με την κλινική ιατρική (findings) και λέξεις που παραλλάσσουν τη σημασία τους (modifiers). Η όλη διαδικασία ξεκινά έχοντας ως είσοδο κάποια κείμενα, και δημιουργώντας έναν πίνα κωδικοποίησης, που αντιστοιχεί σε κάθε δομημένη φράση που έχει ως έξοδο η μέθοδος, έναν κωδικό (unique identifier, UI) από ένα concept μιας οντολογίας του συγκεκριμένου πεδίου, που είναι το πιο κοντινό στη σημασία του συγκεκριμένου finding σε συνδυασμό με τους modifiers που το συνοδεύουν. Επίσης, γίνεται σύνδεση της φράσης μέσα στο κείμενο από την οποία προήλθε η δομημένη έξοδος με τον κωδικό αυτό. Π.χ. για τη φράση “Status post myocardial infraction in 1995” θα εντοπιστεί ο κωδικός C0856742 του UMLS (που αντιστοιχεί στο UMLS concept “post mi”), μαζί με έναν date modifier με τιμή 1995. Σημειώνεται πως θα μπορούσε να του είχε δοθεί ο κωδικός C0027051 (που αντιστοιχεί στο UMLS concept “myocardial infraction”), αλλά δεν είναι τόσο κοντά στη φράση μας όσο το concept “post mi” το οποίο και επιλέγετε.

Χρησιμοποιούν το MedLEE, μέσω του οποίου περνάνε οι προτάσεις ενός κειμένου, και το οποίο εξάγει την πληροφορία στη δομή που θέλουν ώστε να μπορέσει αργότερα να γίνει μια αντιστοίχιση με τον κατάλληλο κωδικό του UMLS. Μια μικρή περιγραφή των συστατικών του MedLEE (Σχήμα 4.18) δίνετε παρακάτω:



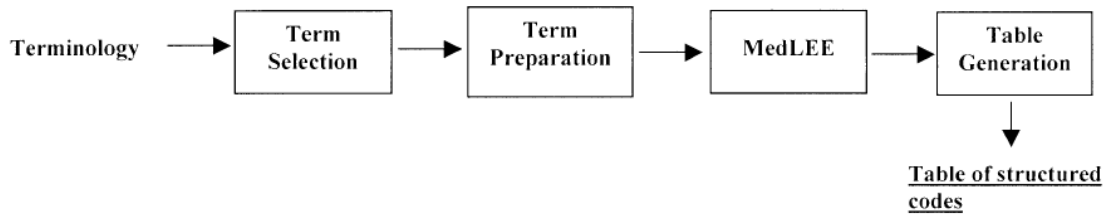
Σχήμα 4.18 Τα Συστατικά Μέρη του MedLEE. [48]

- **Pre-Processor**: Διαχωρίζει το κείμενο σε ενότητες, παραγράφους, προτάσεις και λέξεις, και κάνει λεκτική αναζήτηση για την ανεύρεση και ταξινόμηση λέξεων και φράσεων που αποτελούνται από περισσότερες από μια λέξη (multiword phrases) και εύρεση των κανονικών τους μορφών χρησιμοποιώντας κάποιο λεξικό. Έτσι μια πρόταση "Myocardial infraction in 1995" θα ληφθεί ως μια ακολουθία τριών όρων, "myocardial infraction", "in" και "1995", γιατί ο $1^{ος}$ όρος λαμβάνεται ως multiword phrase στο λεξικό. Ο όρος 'Myocardial infraction" θα ταξινομηθεί ως τύπου "finding", το "in" σαν πρόθεση της Αγγλικής γλώσσας, και το "1995" σαν αριθμός. Ο pre-Processor του MedLEE χειρίζεται επίσης κάποιες συντομογραφίες, χρησιμοποιώντας έναν πίνακα συντομογραφιών, καθώς επίσης κάνει κάποια αποσαφήνιση λέξεων βασιζόμενος σε κάποιους κανόνες συμφραζομένων.
- **Parser**: καθορίζει την αρχική δομημένη μορφή για κάθε πρόταση, χρησιμοποιώντας κάποια γραμματική που περιλαμβάνει συντακτικούς και σημασιολογικούς κανόνες. Η δομή είναι της μορφής λίστας, της οποίας το πρώτο στοιχείο αντιστοιχεί στον τύπο της πληροφορίας, το δεύτερο στην τιμή, και τα υπόλοιπα είναι οι modifiers της τιμής. Π.χ. το "Status postmyocardial infarction in 1995" θα δομηθεί με την εξής μορφή [problem, 'myocardial infraction', [date, '19950000'], [status, post]]. Στο παράδειγμα αυτό βρέθηκε ως primary finding to "myocardial infarction" που έχει έναν modifier χρόνου, τύπου "status" με τιμή "post", και έναν modifier τύπου ημερομηνίας με τιμή "19950000".

- **Error recovery**: το συστατικό αυτό προσπαθεί να ξεκολλήσει την ανάλυση του κειμένου, αν η αρχική προσπάθεια αποτύχει. Αυτό μπορεί να συμπεριλαμβάνει την παράβλεψη λέξεων και τον διαχωρισμό του κειμένου σε μικρότερα κομμάτια.
- **Phrase regularization**: εδώ μπορούν να συντεθούν multiword phrases, μετά το στάδιο της ανάλυσης του κειμένου (parsing), αν η πρόταση περιέχει μια μη συνεχόμενη φράση (π.χ. οι μεμονωμένες λέξεις της φράσης διαχωρίζονται από κάτι άλλο). Για παράδειγμα, η φράση "enlarged spleen" ορίζεται στο λεξικό σαν φράση, έτσι για μια φράση όπως "enlarge spleen noted" θα έχουμε την εξής έξοδο [problem, enlarges spleen, [certainty, 'high certainty']]. Όμως αν είχαμε τη φράση "spleen was enlarged", η έξοδος θα διέφερε γιατί θα προέκυπτε από τις μεμονωμένες λέξεις και όχι από τη φράση (π.χ. [problem, enlarged, [body-loc, spleen], [certainty, 'high certainty']]). Ένας άλλος ρόλος του phrase regularization, είναι η χρήση πληροφορίας του πεδίου (domain knowledge), για να προσθέσει πληροφορία στην έξοδο, που είναι υπονοούμενη στο πεδίο. Π.χ. το "infarct" υποδηλώνει το "myocardial infarction" στον τομέα της καρδιολογίας, αλλά θα μπορούσε να αναφέρετε σε κάτι διαφορετικό σε κάποιο άλλο πεδίο. Η πληροφορία του πεδίου ορίζεται σε έναν πίνακα που δημιουργείται χειροκίνητα από ειδικούς του πεδίου. Στο παραπάνω παράδειγμα, αν μια πρόταση περιλαμβάνει το "infarct", τότε ο όρος θα μετασχηματιστεί σε "myocardial infarction".
- **Encoding**: χρησιμοποιεί έναν πίνακα (coding table) για να προσθέσει κωδικούς, όπως κωδικούς του UMLS, στην έξοδο που προήλθε από το προηγούμενο βήμα. Εδώ γίνεται μια αντιστοίχιση των primary findings με τους αντίστοιχους κωδικούς. Στην έκδοση που περιγράφεται εδώ, οι modifiers δε συνδυάζονται με τα primary findings, έτσι ώστε να βρεθεί ένας ποιο κατάλληλος όρος στην οντολογία. Επίσης, στην έκδοση αυτή, η διαδικασία του encoding είναι χρονοβόρα γιατί πολλές φορές η αντιστοίχιση ενός όρου που προήλθε από το MedLEE με έναν όρο της οντολογίας γίνεται χειροκίνητα.

Πιο πάνω όπως προαναφέρθηκε, περιγράφηκε μια παλιότερη έκδοση του MedLEE. Στην αναθεωρημένη έκδοσή του, αυτό που αλλάζει είναι η διαδικασία της δημιουργίας του coding table, η δομή του και η τεχνική που χρησιμοποιείται για την κωδικοποίηση. Η διαδικασία δημιουργίας του πίνακα είναι πιο περίπλοκη αλλά είναι εντελώς αυτοματοποιημένη και

αποτελείται από τέσσερα βήματα: “term selection”, “term preparation”, “parsing με χρήση του MedLEE”, και παραγωγή του coding table, όπως φαίνετε στο Σχήμα 4.19.



Σχήμα 4.19 Διαδικασία Δημιουργίας του Coding Table. [48]

Αφού έχει δημιουργηθεί ο coding table, χρησιμοποιείται από το MedLEE όπως φαίνετε στο Σχήμα 4.18, εδώ γίνεται αντιστοίχηση των κωδικών του UMLS με τις φράσεις του κειμένου με μια διαδικασία που βασίζεται στο ταίριασμα «δομών» (matching of structures) που είχε ως έξοδο το MedLEE και όχι απλά συμβολοσειρών (matching of strings).

Παρακάτω δίνεται μια περιγραφή, του τρόπου δημιουργίας του coding table:

- **Term selection**: Το βήμα αυτό είναι απαραίτητο γιατί κάποιοι όροι μέσα σε μοντέλα αναπαράστασης γνώσης είναι ακατάλληλα για κείμενα που αφορούν το πεδίο της κλινικής ιατρικής, κάποιοι άλλοι προκαλούν πλεονασμούς, και μερικοί άλλοι είναι αρκετά διφορούμενοι. Έτσι το βήμα αυτό μπορεί να είναι διαφορετικό για διαφορετικά μοντέλα αναπαράστασης γνώσης. Για κωδικοποίηση σύμφωνα με το UMLS, η διαδικασία αυτή αποτελείται από τα εξής: (1) Επιλογή των σημασιολογικών κλάσεων του UMLS, που είναι σχετικές με το κλινικό πεδίο, και επιλογή όρων που σχετίζονται με τις κλάσεις αυτές. (2) Απομάκρυνση όρων που περιλαμβάνουν τη λέξη “other”. (3) Απομάκρυνση όρων που περιλαμβάνουν τα εξής: “nos”, “nec”, “unspecified” και “classified elsewhere”, εάν υπάρχουν αντίστοιχοι όροι χωρίς τις λέξεις αυτές (π.χ. ο όρος “anemia” υφίσταται στο UMLS, έτσι ο όρος “anemia nec” απομακρύνεται). (4) Απομάκρυνση όρων που αναφέρονται στην κτηνιατρική (veterinary), βασιζόμενη σε γνώση που υπάρχει στο SNOMED και (5) απομάκρυνση συντομογραφιών, γιατί αυτές είναι πολύ διφορούμενες και έχει βρεθεί πως προκαλούν πολλά λάθη στην κωδικοποίηση.

- **Term preparation**: Χρησιμοποιείται για την προσθήκη ποικίλων μορφών ενός όρου στον coding table, αν οι μορφές αυτές δεν βρίσκονται ήδη σε αυτόν. Ένα τέτοιο παράδειγμα αφορά όρους που οι λέξεις που τον αποτελούν διαχωρίζονται με κόμμα (π.χ. "infraction, myocardial"). Οι όροι που περιλαμβάνουν κόμμα, κανονικοποιούνται με την απομάκρυνση του κόμμα (π.χ. παράγεται το "infraction myocardial"), και το μέρος της φράσης που ακολουθεί το κόμμα, μεταφέρεται στο αριστερό μέρος (π.χ. παράγεται το "myocardial infraction"). Όταν η διαδικασία του preparation έχει ολοκληρωθεί, δημιουργείται μια λίστα στην οποία ο κάθε όρος συσχετίζεται με έναν κωδικό του UMLS (Unique concept identifiers "CUI"). Το πρώτο μέλος της λίστας είναι το CUI ακολουθούμενο από μια συμβολοσειρά για το concept. Η συμβολοσειρά αυτή είναι ο προτιμούμενος όρος που προτείνει το UMLS. Έτσι όροι που είναι συνώνυμοι σύμφωνα με το UMLS θα ανταποκρίνονται στον ίδιο κωδικό. Στο Σχήμα 4.20 δίνεται ένα παράδειγμα για τον όρο "myocardial infarction".

C0027051^myocardial infarction|myocardial infarction
C0027051^myocardial infarction|heart attack
C0027051^myocardial infarction|myocardial infarction syndrome
C0027051^myocardial infarction|myocardial necrosis
C0027051^myocardial infarction|attack coronary
C0027051^myocardial infarction|necrosis myocardium
C0027051^myocardial infarction|myocardial necrosis syndrome
C0027051^myocardial infarction|coronary thrombosis
C0027051^myocardial infarction|cardiopathy necrotic
C0027051^myocardial infarction|infarction of heart
C0027051^myocardial infarction|infarction, myocardial
C0027051^myocardial infarction|infarction myocardial

Σχήμα 4.20 Η Ανάλυση του όρου "Myocardial Infarction". [48]

- **Parsing**: η διαδικασία αυτή εκτελείται για κάθε όρο που βρίσκεται στον αρχικό πίνακα. Αυτό επιτυγχάνεται λαμβάνοντας υπόψη κάθε όρο σαν μια ολοκληρωμένη πρόταση και περνώντας την σαν είσοδο στο MedLEE για να βρεθεί μια δομημένη μορφή του όρου αυτού. Επιπρόσθετα, αν ο όρος αποτελείται από περισσότερες από μια λέξεις (multiword), τότε θα γίνει parse του όρου και σαν μια ατομική μονάδα,

αλλά και κάθε του λέξη ξεχωριστά. Αυτό γίνεται γιατί μερικές φορές δεν υπάρχει όρος του UMLS που να αντιστοιχίζεται στον multiword όρο, αλλά μπορεί να υπάρχει αντιστοιχία για μια λέξη που περιλαμβάνεται στον όρο. Π.χ. δεν υπάρχει UMLS κωδικός για τον όρο "rash on big toe" αλλά υπάρχει για τον όρο "rash and big toe". Στο παράδειγμα του Σχήματος 4.20 θα γίνει parsing για τον κάθε όρο και η έξοδος που θα βγει για τους τρεις πρώτους όρους ("myocardial infarction", heart attack", και "myocardial infarction syndrome") θα είναι αυτή του Σχήματος 4.21 και ο κάθε όρος θα συσχετιστεί με τον UMLS κωδικό "C0027051^myocardial infraction".

```
[problem,'myocardial infarction']
[problem,infarction,[bodyloc,myocardium]]
[problem,'heart attack']
[problem,attack,[bodyloc,heart]]
[problem,'myocardial infarction',[problemdescr,syndrome]]
[problem,infarction,[bodyloc,myocardium],[problemdescr,
  syndrome]]
```

Σχήμα 4.21 Το Αποτέλεσμα μετά την Επεξεργασία με το MedLEE. [48]

- **Table generation**: Αυτό είναι το τελικό βήμα της όλης διαδικασίας. Στο βήμα αυτό, καταχωρίσεις που παρουσιάζουν προβλήματα στο parsing και που μπορούν να εντοπιστούν αυτόματα, απομακρύνονται. Ένας τύπος προβλήματος είναι ο εντοπισμός ενός όρου του UMLS, που κατηγοριοποιείται στο UMLS σε μια συγκεκριμένη σημασιολογική κατηγορία, αλλά το primary finding που εντοπίστηκε από το MedLEE είναι κάτι διαφορετικό. Σαν παράδειγμα, θα μπορούσε να υπάρξει ένας όρος που στο UMLS κατηγοριοποιείται ως "medical device" ενώ το MedLEE απέδωσε ένα διαφορετικό primary finding, όπως "problem". Το πρόβλημα αυτό υπάρχει λόγο της ύπαρξης ασάφειας σε κάποιον όρο. Προς το παρόν αυτό το πρόβλημα αντιμετωπίζεται με την αναζήτηση σε λεξικά, χρησιμοποιώντας κανόνες συμφραζομένων (context rules) που είναι γραμμένοι με το χέρι.

Αφού τελειώσει και το βήμα αυτό, έχει παραχθεί ο coding table και ένα παράδειγμα μπορούμε να δούμε στο Σχήμα 4.22.

```
C0027051^myocardial infarction| [problem,'myocardial infarction']
C0027051^myocardial infarction| [problem,attack,[bodyloc,heart]]
C0027051^myocardial infarction| [problem,'myocardial infarction',
    [problemdescr,syndrome]]
C0856742^post mi'|[problem,'myocardial infarction',[status,post]]
C0155668^myocardial infarction old|[problem,'myocardial infarction',[status,previous]]
C0155668^myocardial infarction old|[problem,'heart attack',[status,previous]]
C0340293^myocardial infarction anterior|[problem,'myocardial infarction',
    [region,anterior]]
C0746711^myocardial infarction anterior non q wave|[problem,'myocardial infarction',
    [region,anterior],[descriptor,'non q wave']]
```

Σχήμα 4.22 Ο Τελικός Coding Table. [48]

Στη συνέχεια χρησιμοποιείται το MedLEE για να γίνει parsing των προτάσεων κλινικών κειμένων. Η δομημένη έξοδος που παράγει περιέχει τα primary findings και τους modifiers που συσχετίζονται με τα findings. Το βήμα κωδικοποίησης αποτελείτε από το ταίριασμα της δομημένης εξόδου που παράχθηκε με το parsing των προτάσεων, με τις δομημένες εξόδους που έχουν εισαχθεί στον coding table. Το πιο κοντινό ταίριασμα θεωρείται αυτό που θα ταιριάξει σωστά τα primary findings και όσους περισσότερους modifiers γίνεται. Για παράδειγμα θα βρει πως το πιο καλό ταίριασμα για τη δομή [problem, 'myocardial infraction', [date, 19950000], [status, post]]. που εξάχθηκε από το parsing της πρότασης "Status post myocardial infraction in 1995, θα είναι η δομή [problem, 'myocardial infraction', [status, post]] που βρίσκεται στον coding table και συσχετίζεται με τον κωδικό C0856742 και αντιστοιχεί στον UMLS όρο "post mi". Σε κάποιες περιπτώσεις μπορεί να γίνει συσχέτιση με πάνω από έναν κωδικό.

Όταν ένας κωδικός βρεθεί, τότε προστίθεται στο primary finding σαν ένας modifier με όνομα "umls". Πχ. [problem, 'myocardial infraction', [status, post], [region, anterolateral], [umls, C0262564^anterolateral myocardial infract], [umls, C0856742^post mi]].

Η τελική μορφή είναι σε XML, και είναι λίγο πιο περίπλοκη , γιατί περιέχει συνδέσμους στις φράσεις του αρχικού κειμένου, που έτσι συσχετίζονται με τον κατάλληλο κωδικό του UMLS.

ΚΕΦΑΛΑΙΟ 5. ΕΡΓΑΛΕΙΑ ΚΑΙ ΓΛΩΣΣΕΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ ΓΙΑ ΤΗΝ ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ

5.1. Αντιστοίχηση Όρων και Φράσεων στο UMLS (MetaMap – MMTx)

.Το MetaMap [73] είναι ένα εργαλείο το οποίο αντιστοιχεί σε αυθαίρετο κείμενο, τα Concepts του UMLS Metathesaurus που του αναλογούν, ή ισοδύναμα, ανακαλύπτει Concepts του UMLS μέσα σε κείμενο. Το MMTx (MetaMap Transfer) είναι μια προσπάθεια να γίνει το MetaMap διαθέσιμο σε βιοϊατρικούς ερευνητές σαν ένα γενικό, εύκολα παραμετροποιήσιμο περιβάλλον

Το MMTx είναι διαθέσιμο και σαν μια εφαρμογή java για άμεση χρήση του από κάποιον χρήστη, αλλά και σαν ένα Java API ώστε οι λειτουργίες του να μπορούν να ενσωματωθούν σε άλλες Java εφαρμογές που έχουν την ανάγκη να κάνουν αντιστοιχία κειμένου στα αντίστοιχα Concepts του UMLS που το εκφράζουν.

Σαν μέρος αυτής της διαδικασίας αντιστοίχησης (κειμένου σε Concepts), το MMTx τεμαχίζει το κείμενο σε ενότητες (sections), προτάσεις (sentences), φράσεις (phrases), όρους (terms) και λέξεις (words). Στη συνέχεια κάνει αντιστοίχιση μεταξύ των καταλληλότερων Concepts

του UMLS και φράσεις του κείμενου που του δόθηκε και έχουν σαν βασικό συστατικό τους κάποιο ουσιαστικό (Noun Phrases).

Παρακάτω θα δώσουμε μια ανάλυση του τρόπου με τον οποίο δουλεύει το MetaMap, μια περιγραφή της διαδικασίας που ακολουθεί το MMTx, καθώς και των κλάσεων που χρησιμοποιούνται στο Java API του.

5.1.1. Το MetaMAP

5.1.1.1. Η Βασική Στρατηγική Αντιστοίχησης του MetaMap (The Basic Mapping Strategy)

1) Κάνε ένα πέρασμα του κειμένου (parsing) και χώρισε το σε φράσεις με βασικό χαρακτηριστικό τους ένα ουσιαστικό (noun phrases). Για κάθε φράση εκτέλεσε τα υπόλοιπα βήματα.
2) Παρήγαγε τις παραλλαγές για κάθε noun phrase, όπου κάθε παραλλαγή αποτελείται από μία ή περισσότερες λέξεις της noun phrase με όλες τις ορθογραφικές ποικιλομορφίες τους (spelling variants), τις συντομογραφίες τους (abbreviations), τα ακρωνύμια τους (acronyms), τα συνώνυμά τους (synonyms), ποικιλομορφίες που προέρχονται από την κλίση τους (inflectional variants) και από την ετυμολογία τους (derivational variants) και συνδυασμούς αυτών που έχουν νόημα.
3) Δημιούργησε το υποψήφιο σύνολο που περιέχει όλα τα Strings του Metathesaurus που περιέχουν μία από τις ποικιλομορφίες του παραπάνω βήματος.
4) Για κάθε υποψήφιο string, υπολόγισε το κατά πόσο κατάλληλη είναι η αντιστοίχισή του σε μια noun phrase κάνοντας χρήση μιας συνάρτησης αποτίμησης.
5) Κάνε συνδυασμούς των υποψήφιων όρων, που εμπλέκονται σε μη συνεκτικά κομμάτια της noun phrase, και επαναϋπολόγισε την καταλληλότητα της αντιστοίχισης, και επέλεξε αυτούς που έχουν το υψηλότερο σκορ να αποτελέσουν το σύνολο των καλύτερων αντιστοιχίσεων μεταξύ της noun phrase και του Metathesaurus.

Παρακάτω δίνετε μια περιγραφή των βημάτων 2-5 της στρατηγικής αντιστοίχησης.

5.1.1.2. Παραλλαγές μιας noun phrase (Noun Phrase Variants)

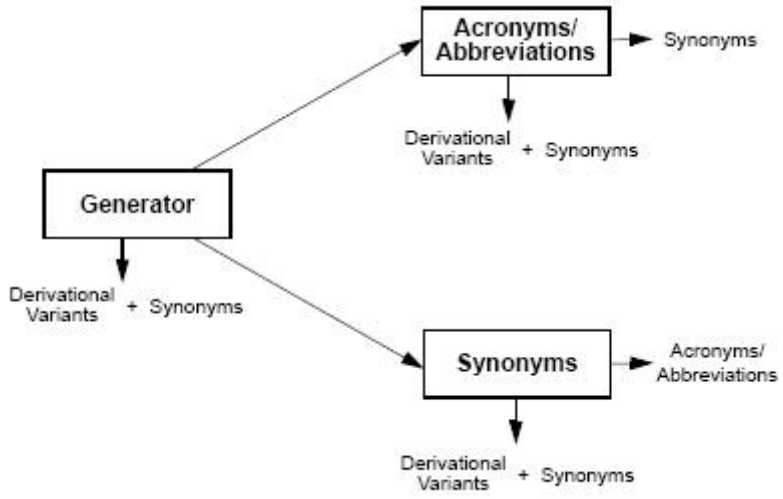
Η διαδικασία αντιστοίχησης ξεκινά με τον υπολογισμό ενός συνόλου κάποιον γεννητόρων (variant generators) για κάθε noun phrase που εντόπισε ο αναλυτής του κειμένου (parser). Ένας γεννήτορας είναι κάθε υπο-ακολουθία λέξεων στη φράση που έχει νόημα, και μια υπο-ακολουθία έχει νόημα, όταν είναι είτε μια απλή λέξη είτε υπάρχει στο SPECIALIST lexicon (το SPECIALIST lexicon, είναι ένα λεξικό όρων του UMLS). Για παράδειγμα οι γεννήτορες για τη noun phrase "liquid crystal thermography" είναι οι εξής: "liquid crystal thermography", "liquid crystal", "liquid", "crystal" και "thermography", ενώ για τη φράση "ocular complications" παίρνουμε τους γεννήτορες "ocular" και "complications".

Οι ποικιλομορφίες υπολογίζονται για κάθε γεννήτορα σύμφωνα με το Σχήμα 5.1. Η επεξεργασία του εκάστοτε γεννήτορα προχωράς ως εξής:

1) Υπολόγισε τα ακρωνύμια, συντομογραφίες και συνώνυμα για κάθε γεννήτορα.
2) Εμπλούτισε τα τρία παραπάνω σύνολα, υπολογίζοντας τις ποικιλομορφίες μέσω ετυμολογίας (derivational variants) , και τα συνώνυμα αυτών.
3) Για κάθε μέλος του συνόλου Acronyms/Abbreviations, υπολόγισε τα συνώνυμά του.
4) Για κάθε μέλος του συνόλου Synonyms, υπολόγισε τα Acronyms/Abbreviations που του αναλογούν.

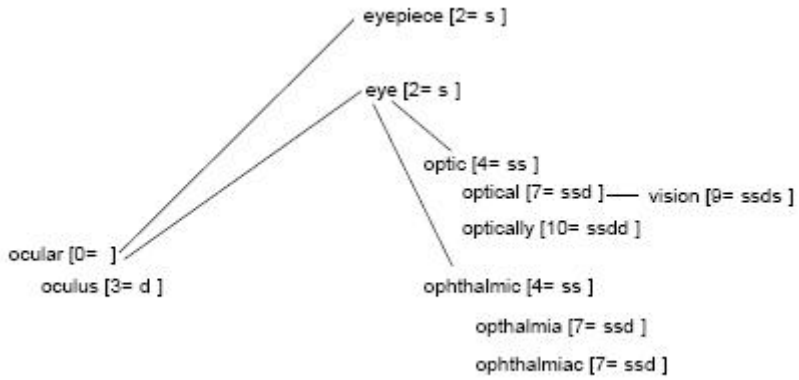
Τα Acronyms και οι Abbreviations δεν υπολογίζονται αναδρομικά, γιατί αυτό επιφέρει σχεδόν πάντα λανθασμένα αποτελέσματα.

Οι Derivational Variants και τα Synonyms υπολογίζονται αναδρομικά, αφού αυτό επιφέρει συνήθως ποικιλομορφίες που έχουν νόημα.



Σχήμα 5.1 Σχηματική Αναπαράσταση Υπολογισμού Ποικιλομορφιών. [73]

Οι ποικιλομορφίες που υπολογίστηκαν για τον γεννήτορα “ocular” φαίνονται στο Σχήμα 5.2. Κάθε ποικιλομορφία, ακολουθείται από ένα distance score, που είναι ένα μέτρο για το κατά πόσο διαφοροποιείται από τον γεννήτορα, και ένα ιστορικό που δείχνει την πορεία που ακολουθήθηκε κατά τον υπολογισμό της. Για παράδειγμα το “optical” (με distance score “9” και history “ssd”) είναι μια derivational variant ενός συνωνύμου (optic), ενός συνωνύμου (eye), του “ocular”.



Σχήμα 5.2 Οι Ποικιλομορφίες του Όρου “Ocular”. [73]

Η παραγωγή των παραπάνω ποικιλομορφιών, γίνετε με τη χρήση των εξής πηγών γνώσης:

1) Του SPECIALIST lexicon και ενός πίνακα με κανονικές μορφές που προήλθαν από αυτόν.
2) Μια βάση γνώσης του SPECIALIST που περιέχει ακρωνύμια και συντομογραφίες (acronyms, abbreviations).
3) Μια βάση γνώσης του SPECIALIST που περιέχει κανόνες για την παραγωγή των derivational variants.
4) Δυο βάσεις συνωνύμων.

5.1.1.3. Οι Υποψήφιοι Όροι του METATHESAURUS (Metathesaurus Candidates)

Οι υποψήφιοι όροι του Metathesaurus για μια noun phrase είναι το σύνολο όλων των strings του Metathesaurus που περιέχουν τουλάχιστον μια ποικιλομορφία που έχει υπολογιστεί για τη φράση. Οι υποψήφιοι εντοπίζονται εύκολα με τη χρήση ενός ευρετηρίου από λέξεις προς όλα τα stings του Metathesaurus που τις περιέχουν. Οι υποψήφιοι όροι για τη φράση "ocular complications" δίνονται στο σχήμα 5..3. Μαζί με τα stings που επιστρέφονται επιστρέφεται και ο προτιμώμενος όρος που χρησιμοποιεί το Metathesaurus για την ονομασία του εκάστοτε Concept στο οποίο ανήκει το κάθε string.

861 Complications (Complication)
861 complications <1>
638 Eye
611 Optic (Optics)
588 Ophthalmia (Endophthalmitis)

Σχήμα 5.3 Οι Υποψήφιοι Όροι για τη Φράση "Ocular Complications". [73]

Οι υποψήφιοι όροι ταξινομούνται σύμφωνα με τη συνάρτηση αποτίμησης που περιγράφεται παρακάτω. Στο συγκεκριμένο παράδειγμα οι καταλληλότεροι υποψήφιοι όροι, είναι οι "Complications" και "complications <1>" που προήλθαν μέσω της επεξεργασίας που έγινε στην κεφαλή της noun phrase (head of noun phrase). Οι υπόλοιποι υποψήφιοι όροι είναι ποικιλομορφίες του "ocular" και ταξινομούνται σύμφωνα με την ομοιότητά τους στο "ocular".

5.1.1.4. Η Συνάρτηση Αποτίμησης (The Evaluation function)

Η συνάρτηση αποτίμησης υπολογίζει ένα μέτρο για την ποιότητα της αντιστοίχησης μεταξύ της φράσης και ενός υποψηφίου όρου. Η συνάρτηση αυτή είναι βασισμένη σε τέσσερα συστατικά: κεντρικότητα (centrality), απόκλιση (variation), κάλυψη (coverage) και συνεκτικότητα (cohesiveness). Μια κανονικοποιημένη τιμή μεταξύ 0 (η χειρότερη αντιστοιχία) και 1 (η καλύτερη αντιστοιχία) υπολογίζεται για κάθε ένα από τα παραπάνω συστατικά της συνάρτησης. Ένας σταθμικός μέσος (weighted average) υπολογίζεται, στον οποίο η coverage και η cohesiveness λαμβάνουν διπλάσιο βάρος από ότι η centrality και η variation. Το αποτέλεσμα στη συνέχεια κανονικοποιείται σε μια τιμή μεταξύ 0 (δεν βρέθηκε αντιστοιχία) και 1000 (βρέθηκε ακριβώς ο ίδιος όρος). Αν το MetaMap τεθεί να μην λαμβάνει υπόψη τη σειρά των λέξεων στη φράση, το συστατικό "coverage" αντικαθίσταται από το συστατικό "involvement". Παρακάτω περιγράφονται τα συστατικά της συνάρτησης αποτίμησης.

- **Centrality**: Η τιμή του centrality είναι 1 αν το string του υποψήφιου όρου εμπλέκει την κεφαλή της φράσης και 0 διαφορετικά. Για τη Noun phrase "ocular complications", ο υποψήφιος όρος "Complications" έχει τιμή για το centrality 1, ενώ ο υποψήφιος όρος "Eye" έχει τιμή centrality 0.
- **Variation:** Η τιμή αυτή είναι μια εκτίμηση του κατά πόσο η ποικιλομορφία που έχει υπολογιστεί για κάποιον γεννήτορα, διαφέρει από τον όρο που βρίσκεται στη φράση. Υπολογίζεται με τον προσδιορισμό αρχικά της variation distance για κάθε ποικιλομορφία στο Metathesaurus string. Η απόσταση αυτή είναι το άθροισμα της απόστασης για κάθε βήμα που γίνεται μέχρι την παραγωγή της ποικιλομορφίας. Η απόσταση αυτή δίνεται στο Σχήμα 5.4.

Variant Type	Distance Value
spelling	0
inflectional	1
synonym or acronym/abbreviation	2
derivational	3

Σχήμα 5.4 Τρόπος Εκτίμησης της Απόστασης μιας Ποικιλομορφίας από τον Όρο της Φράσης. [73]

Η variation distance καθορίζει την τιμή απόκλισης για τη συγκεκριμένη ποικιλομορφία σύμφωνα με τον τύπο V=4/(D+4). Όσο η συνολική τιμή απόστασης (distance value), D, αυξάνει από το 0, το V μειώνεται από το άνω φράγμα του το 1, και φράσσεται κάτω από το 0. Η τελική τιμή απόκλισης (variation value) για τον υποψήφιο όρο είναι ο μέσος όρος των Distance values της κάθε ποικιλομορφίας. Για το "ocular complications", το "Eye" έχει variant distance 2 και έτσι έχει variation value 2/3=(4/(2+4)). Το "complications" έχει variant distance 0 και έτσι η variation value του θα είναι 1.

- **Coverage**: Η τιμή αυτή δεικτοδοτεί το κατά πόσο το sting του Metathesaurus και η noun phrase συντελούν στο μεταξύ τους ταίριασμα. Με σκοπό να υπολογιστεί η τιμή αυτή, καταμετράται το πλήθος των λέξεων που συμμετέχουν στο ταίριασμα και από το Metathesaurus string και από τη φράση. Τα πλήθη αυτά ονομάζονται, "Metathesaurus span" και "phrase span" αντίστοιχα. Η Coverage value για το Metathesaurus string είναι το Metathesaurus span διαιρεμένο με το μήκος του sting. Όμοια, η Coverage value για τη φράση είναι το phrase span διαιρεμένο με το μήκος της φράσης. Η τελική coverage value, είναι ο σταθμικός μέσος όρος των τιμών του Metathesaurus string και της φράσης, όπου στο Metathesaurus string δίνεται το διπλάσιο βάρος από ότι στη φράση. Για παράδειγμα στη noun phrase "ocular complications" και είτε το "Eye" είτε το "Complications" για Metathesaurus string, τα Metathesaurus span και phrase span είναι και τα δύο ίσα με 1, και η coverage value θα είναι 5/6=(2/3*(1/1)+1/3*(1/2)).
- **Cohesiveness**: Η cohesiveness value είναι όμοια με την coverage value, αλλά δίνει έμφαση στη σημαντικότητα των συνδεδεμένων συστατικών. Συνδεδεμένο συστατικό θεωρείται η μέγιστη ακολουθία συνεχόμενων λέξεων που συμμετέχουν στην αντιστοίχηση. Υπολογίζονται τα συνδεδεμένα συστατικά για το Metathesaurus string και τη φράση. Αυτό παράγει ένα σύνολο από μεγέθη των connected components και για το Metathesaurus string και για τη φράση. Για παράδειγμα για τη φράση "sleep obstructive apnea" και το υποψήφιο concept "Sleep Apneas", θα είχαμε για connected component sizes τα [[1,1],[2]], το οποίο δείχνει πως η φράση έχει δύο connected components που το καθένα έχει μέγεθος 1 (τα "sleep" και "apnea"), ενώ ο υποψήφιος όρος έχει ένα μοναδικό connected component μεγέθους 2 (το "sleep apneas"). Η cohesiveness value για το Metathesaurus string, είναι το άθροισμα των τετραγώνων των connected component sizes του Metathesaurus string διαιρεμένο με το τετράγωνο

του μήκους του string. Όμοια υπολογίζεται και η cohesiveness value για τη φράση. Η τελική cohesiveness value είναι ο σταθμικός μέσος όρος των cohesiveness values του Metathesaurus string και της φράσης, όπου και πάλι η cohesiveness value του metathesaurus string έχει διπλάσιο βάρος από αυτή της φράσης. Για το "ocular complications" και είτε το "Eye" είτε το "Complications", τα connected components sizes και για το Metathesaurus string και για τη φράση είναι [1], αφού μία μόνο λέξη και από τα δυο συμμετέχει στην αντιστοίχηση. Έτσι η cohesiveness value θα είναι $3/4=(2/3*(1^2/1^2)+1/3*(1^2/2^2))$.

Τελικά η αποτίμηση για τον υποψήφιο όρο "Eye" είναι ο σταθμισμένος μέσος (0+2/3+2*(5/6)+2*(3/4))/6, το οποίο πολλαπλασιάζοντας το με το 1000, παίρνουμε την αποτίμησή του σε μια κλίμακα μεταξύ 0 και 1000 που είναι ίσο με 638.

- **Involvement**: Η τιμή αυτή αντικαθιστά την coverage value, όταν αγνοείται η σειρά των λέξεων. Η involvement value για τη φράση, είναι η αναλογία των λέξεων της φράσης που μπορούν να αντιστοιχηθούν σε λέξεις του Metathesaurus string χωρίς να λαμβάνεται υπόψη η διάταξη των λέξεων. Για παράδειγμα δοσμένης της φράσης "Advanced cancer of the lung" με τις λέξεις [advanced, cancer, lung], και του Metathesaurus string "Lung Cancer" με τις λέξεις [lung,cancer], λαμβάνοντας υπόψη τη διάταξη των λέξεων, θα γινόταν αντιστοίχηση του lung στο lung, αλλά όχι του cancer, γιατί προηγείται στη διάταξη του lung στη φράση. Έτσι η involvement value για τη φράση θα είναι εδώ 2/3, αντίθετα με την coverage value που θα ήταν 1/3. Το αντίστοιχο γίνεται και για την involvement value του Metathesaurus string, η οποία θα έχει τιμή στο συγκεκριμένο παράδειγμα ίση με 2/2=1 αντί του ½ που θα είχε για coverage value. Έτσι η τελική involvement value για το παράδειγμά μας θα είναι ο σταθμισμένος μέσος όρος (2/3+1)/2=0.83.

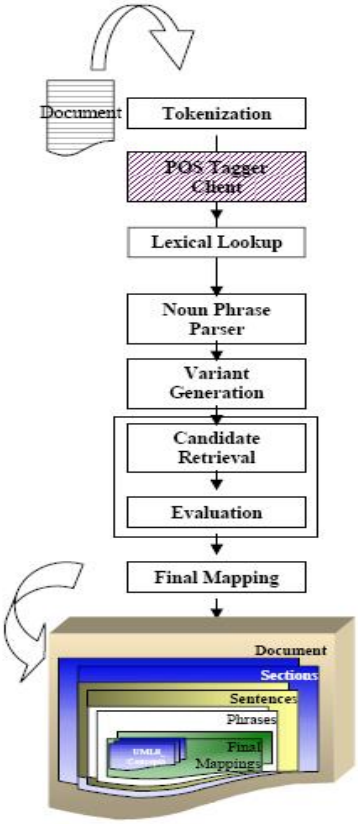
5.1.1.5. Η Τελική Αντιστοίχιση (The Final Mapping)

Το τελικό βήμα στον αλγόριθμο αντιστοίχησης αποτελείται από την εξέταση συνδυασμών των υποψήφιων όρων του Metathesaurus, που συμμετέχουν στην αντιστοίχηση με μη συνεκτικά μέρη της noun phrase. Η συνάρτηση αποτίμησης εφαρμόζεται στους

συνδυασμένους υποψήφιους όρους, και επιστρέφεται το καλύτερο ταίριασμα για την τελική αντιστοίχηση.

5.1.2. Το MMTx

5.1.2.1. Η Λειτουργία του MMTx Συνοπτικά



Σχήμα 5.5 Πρώτο Σχήμα του Πρώτου Κεφαλαίου.

Το MMTx (Σχήμα 5.5) περνάει το κείμενο, που είναι προς επεξεργασία, στον tokenizer ο οποίος το χωρίζει (tokenize) σε ενότητες (sections) που περιέχουν προτάσεις (sentences), οι οποίες προτάσεις περιέχουν λέξεις (tokens).

Οι προτάσεις (sentences) μπορούν να περάσουν σε έναν "part of speech tagger (POS tagger)" ο οποίος μπορεί να αναθέσει στην κάθε λέξη (token) το μέρος του λόγου στο οποίο

αντιστοιχεί στην εκάστοτε πρόταση. Το MMTx δεν περιέχει κάποιον ενσωματωμένο POS tagger, αλλά μπορεί να δώσει πρόσβαση σε εξωτερικούς μέσω κάποιων programming interfaces.

Τα word tokens της κάθε πρότασης, περνάνε από μια διαδικασία αντιστοίχισης με όρους του SPECIALIST Lexicon στο Lookup συστατικό του συστήματος, ώστε να γίνει συνδυασμός των word tokens σε multi-word όρους και για να εντοπιστούν τα μέρη του λόγου στα οποία ανήκουν. Το αποτέλεσμα είναι λεκτικά στοιχεία (lexical elements) που αποτελούνται από word tokens.

Στη συνέχεια οι προτάσεις (sentences) μπαίνουν ως είσοδος σε έναν noun phrase parser για να διαχωριστούν (tokenize) σε φράσεις (phrases). Ο noun phrase parser κάνει χρήση των μερών του λόγου που έχουν δοθεί στα lexical elements από το προηγούμενο συστατικό του συστήματος (Lexical Lookup) και των μερών του λόγου που δόθηκαν από τον POS tagger, αν χρησιμοποιήθηκε. Το αποτέλεσμα είναι φράσεις που αποτελούνται από lexical elements. Οι φράσεις αυτές είναι τοποθετημένες (συνδεδεμένες) με τις προτάσεις (sentences) από τις οποίες προήλθαν.

Ποικιλομορφίες, συμπεριλαμβανομένων των συνωνύμων (synonyms), ορθογραφικών ποικιλομορφιών (spelling variants), ποικιλομορφιών που προέρχονται από την ετυμολογία (derivations), ποικιλομορφιών που προέρχονται από την κλίση (inflections), ακρωνυμίων (acronyms), συντομογραφιών (abbreviations), τις επεκτάσεις των ακρωνυμίων και των συντομογραφιών και όρων που προκύπτουν από τον αναδρομικό συνδυασμό όλων των προηγούμενων, ανακτώνται για τις λέξεις (words) και τα λεκτικά στοιχεία (lexical elements) της κάθε φράσης στο Variant Generation συστατικό του συστήματος. Το αποτέλεσμα αυτού του σταδίου είναι φράσεις (phrases) που περιέχουν ποικιλομορφίες. Κάθε ποικιλομορφία έχει ένα κόστος (cost) ή απόσταση (distance) που δεικτοδοτεί το πόσες τροποποιήσεις γίνανε από την αρχική μορφή ενός όρου για να προκύψει η ποικιλομορφία.

Οι φράσεις και οι ποικιλομορφίες τους, χρησιμοποιούνται για να ανακτηθούν τα Strings από το UMLS με τα οποία αντιστοιχούνται. Το σύνολο των strings του UMLS που ταιριάζουν σε μια φράση, ή ποικιλομορφία της φράσης, ονομάζονται υποψήφιοι όροι (candidates).

Οι υποψήφιοι όροι περνάνε μια διαδικασία αποτίμησης για το κατά πόσο καλά ταιριάζουν με την εκάστοτε φράση. Η διαδικασία αυτή είναι βασισμένη σε αρκετά κριτήρια που περιγράφηκαν στην ενότητα που παρουσιαζόταν το MetaMap. Το αποτέλεσμα αυτού του βήματος είναι η βαθμολόγηση των υποψήφιων όρων σε μια κλίμακα μεταξύ 0 και 1000, με το 1000 να δεικτοδοτεί ένα ακριβές ταίριασμα.

Για κάθε υποψήφιο όρο, συγκεντρώνονται το Concept του και ο σημασιολογικός του τύπος. Το αποτέλεσμα της διαδικασίας παραγωγής υποψηφίων όρων και της αποτίμησής τους, είναι ένα σύνολο από UMLS_Concept_Pointers που συσχετίζονται με την κάθε φράση. Κάθε UMLS_Concept_Pointer περιλαμβάνει ένα evaluation score, ένα σύνολο από UMLS_String_Pointer και ένα σύνολο από UMLS_Semantic_Type_Pointer.

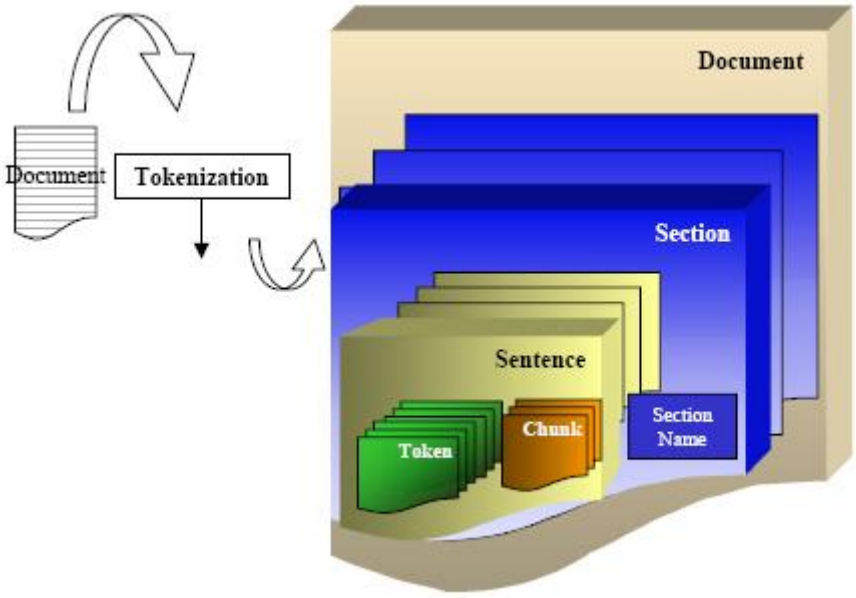
Αν η φράση δεν καλύπτεται πλήρως από κάποιο υποψήφιο UMLS String, γίνονται συνδυασμοί των υποψήφιων όρων, ώστε να καλύπτουν όσο το δυνατόν καλύτερα τη φράση στο Final Mapping συστατικό του συστήματος. Το αποτέλεσμα του βήματος αυτού είναι ένα σύνολο από Final_Mapping. Κάθε Final_Mapping περιλαμβάνει το σύνολο των UMLS_Concept_Pointer που συνδέονται με τα Concepts που καλύπτουν καλύτερα τη φράση. Κάθε Final_Mapping περιλαμβάνει επίσης ένα final mapping score.

5.1.2.2. Κλάσεις και Διαδικασίες του MMTx API (Container Classes and Processes)

Το MMTxAPI είναι ο συνιστώμενος τρόπος για την ενσωμάτωση του MMTx σε άλλες εφαρμογές. Σε αυτό είναι ορισμένες, μέθοδοι που αντιστοιχούν κείμενο από ένα ολόκληρο έγγραφο, έναν απλό όρο και πολλά άλλα σε όρους του UMLS. Το βασικό μέρος του API είναι μια κλάση, δημιουργείται ένα instance αυτής και περιέχει μεθόδους που καλούνται ξανά και ξανά. Από τη στιγμή που έχει δημιουργηθεί ένα τέτοιο instance, αυτό παίρνει ως είσοδο είτε ένα container όπως ένα είδη υπάρχον κείμενο (Document), ενότητα (Section) ή πρόταση (Sentence) είτε ένα κείμενο. Οι μέθοδοι που λαμβάνουν μια container class σαν είσοδο, προσθέτουν στο ίδιο αυτό container. Οι μέθοδοι που λαμβάνουν κείμενο σαν είσοδο επιστρέφουν ένα container instance. Οι container classes περιλαμβανομένων των Document, Sentence και Phrase θα καλυφθούν με μεγαλύτερη λεπτομέρεια παρακάτω.

Παρακάτω θα περιγραφούν οι container classes του MMTx, δίνοντας πως το MMTx τις χρησιμοποιεί και τις συμπληρώνει (populated) όταν καλείται μια από τις μεθόδους processDocument, processSentence, processString και processTerm.

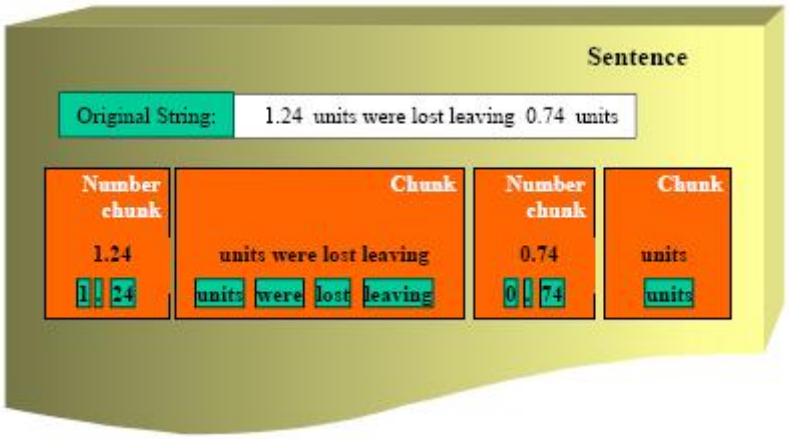
Οι Container Classes Document, Section και Sentence



Σχήμα 5.6 Αναπαράσταση των Κλάσεων Document, Section και Sentence σε σχέση με τη Δομή του Κειμένου. [73]

Κατά τη διάρκεια του tokenization, το κείμενο προς επεξεργασία μετατρέπεται σε ένα instance τύπου Document class. Ένα Document χωρίζεται μετέπειτα σε sections. Όταν έχουμε να κάνουμε με αδόμητα έγγραφα, τα sections είναι παράγραφοι. Όταν έχουμε να κάνουμε με δομημένα έγγραφα (όπως με MEDLINE Citations), κάθε section του citation, όπως το Title section, το Author section, το abstract section, αντιστοιχεί σε ένα ξεχωριστό section. Τα section μπορούν να έχουν κάποια ετικέτα (labeled). Για παράδειγμα μπορεί να είναι βολικό να γνωρίζουμε πως κάποιο συγκεκριμένο section είναι τύπου title section. Κάθε section περιλαμβάνει ένα σύνολο από προτάσεις (sentences). Κάθε Sentence περιλαμβάνει ένα πλήθος από containers αλλά μόνο τα chunks και tokens (word) containers γεμίζονται με πληροφορία κατά την διαδικασία του tokenization. Η Παραπάνω διαδικασία σε σχέση με τις κλάσεις που χρησιμοποιούνται αναπαρίσταται στο Σχήμα 5.6.

Οι Container Classes Sentence, Chunk και Token



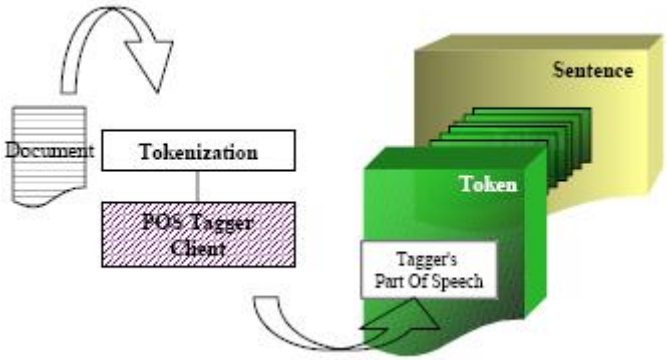
Σχήμα 5.7 Αναπαράσταση των Κλάσεων Sentence, Chunk και Token σε Σχέση με τη Δομή του Κειμένου. [73]

Κάθε sentence υφίσταται περεταίρω επεξεργασία από τον tokenizer, ώστε να βρεθούν κομμάτια (chunk) κειμένου, που μπορούν να αναγνωριστούν σαν μια συνεχόμενη μονάδα μέσω ενός λογισμικού αναγνώρισης προτύπων. Τα Chunks περιλαμβάνουν ημερομηνίες (dates), υπερσυνδέσεις (urls), emails, πραγματικούς αριθμούς (real numbers) και συναφή πρότυπα. Τα Chunks προ το παρόν αναγνωρίζονται από ένα μικρό σύνολο κανονικών εκφράσεων. Τα chunks που αναγνωρίζονται παίρνουν μια ετικέτα (label). Το κείμενο γύρω από τα labeled chunks επίσης ομαδοποιείται σε chunks, αλλά χωρίς ετικέτα (unlabeled chunks). Κάθε chunk στη συνέχεια διαχωρίζεται (tokenized) σε word tokens. Τα labeled chunks γίνονται lexicalElements κατά τη διαδικασία του Lexical Lookup και δεν αναζητούνται στο SPECIALIST Lexicon.

Στο παραπάνω παράδειγμα (Σχήμα 5.7), οι προτάσεις έχουν διαχωριστεί σε τέσσερα chunks, δύο labeled και δύο unlabeled. Τα chunks όπως φαίνεται περιέχουν τα αρχικά instances των tokens που συσχετίζονται με την πρόταση.

Τα chunks είναι ένα προσωρινό container που δεν γίνεται ξανά αναφορά σε αυτό μετά τη δημιουργία των lexicalElements.

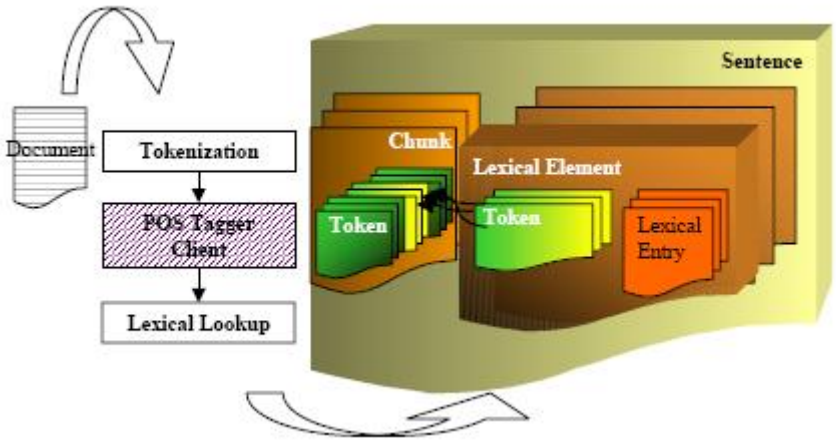
Η Container Class Token



Σχήμα 5.8 Αναπαράσταση της Κλάσης Token σε Σχέση με τη Δομή του Κειμένου. [73]

Αν χρησιμοποιηθεί ένας POS Tagger, η διαδικασία POS Tagger client θα αναθέσει σε κάθε token της κάθε πρότασης (sentence) ένα μέρος του λόγου (part of speech). Αυτή η διαδικασία είναι χρήσιμη κατά τη διάρκεια της αναγνώρισης των noun phrases, για την αποσαφήνιση λέξεων στις οποίες μπορούν να αποδοθούν πολλαπλά μέρη του λόγου. Μια αναπαράσταση της διαδικασία φαίνεται στο Σχήμα 5.8.

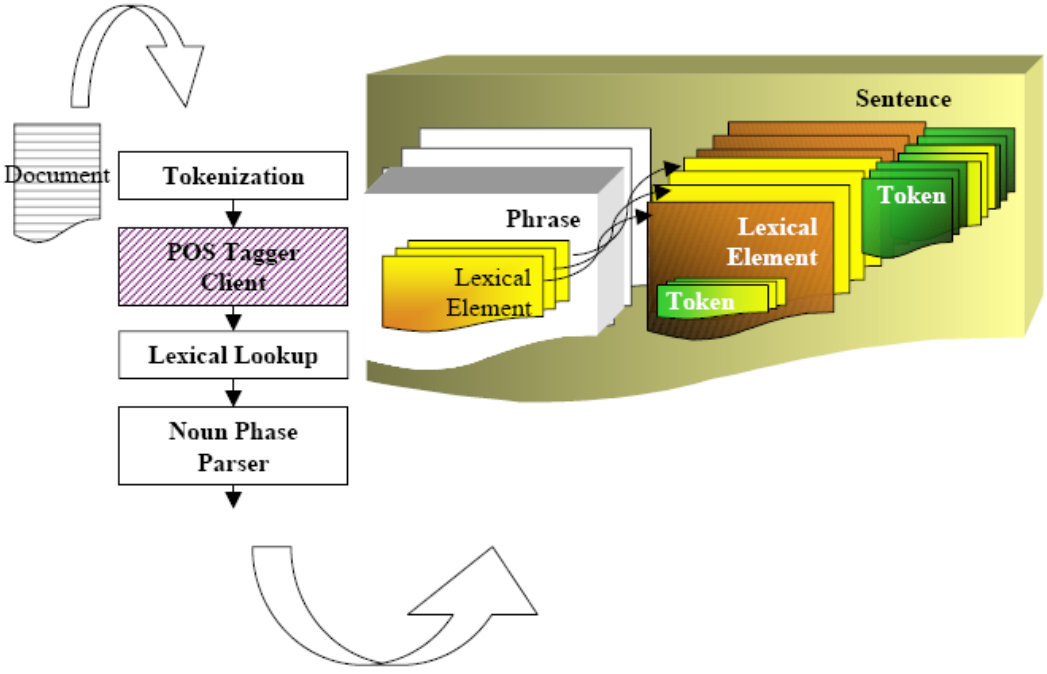
Οι Container Classes LexicalElement και LexicalEntry



Σχήμα 5.9 Αναπαράσταση των Κλάσεων LexicalElement και LexicalEntry σε Σχέση με τη Δομή του Κειμένου. [73]

Η Lexical Lookup διαδικασία, διατρέχει τα chunks. Τα tokens από κάθε labeled chunk γίνονται τα tokens ενός νέου LexicalElement. Τα LexicalElements που δημιουργούνται από labeled chunks περιλαμβάνουν ημερομηνίες (dates), αριθμούς (numbers) και χρήματα (money). Τα tokens των unlabeled chunks χρησιμοποιούνται για την αναγνώριση και το ταίριασμα με όρους πολλαπλών λέξεων (multi-word terms) από το SPECIAL Lexicon. Οι όροι με τους οποίους γίνεται αντιστοίχηση σε αυτό το στάδιο δημιουργούν νέα LexicalElement instances που περιλαμβάνουν τα tokens που σχηματίζουν τον όρο. Τα LexicalElements προστίθενται σε κάθε πρόταση (sentence) κατά τη διαδικασία του Lexical Lookup. Τα Lexical Elements που προέρχονται από το SPECIALIST Lexicon περιλαμβάνουν επιπλέον συντακτική πληροφορία από το Lexicon σε ένα container που κρατά LexicalEntries. Κάθε LexicalEntry αντιστοιχεί σε μια καταχώριση στο SPECIALIST Lexicon. Τα tokens που περιλαμβάνονται σε κάθε LexicalElement δεν αποτελούν νέα instances, αλλά είναι αναφορές στα tokens που δημιουργήθηκαν κατά τη διαδικασία του tokenization. Στο Σχήμα 5.9 βλέπουμε μια αναπαράσταση των κλάσεων LexicalElement και LexicalEntry.

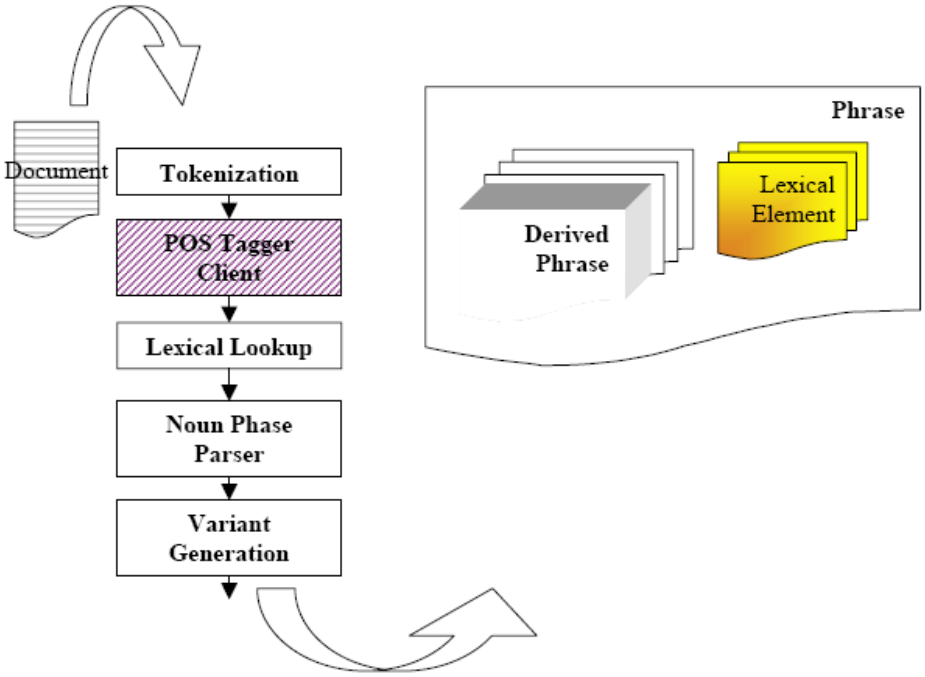
Η Container Class Phrase



Σχήμα 5.10 Αναπαράσταση της Κλάσης Phrase σε Σχέση με τη Δομή του Κειμένου. [73]

Η διαδικασία Noun Phrase Parser, συνδυάζει τα lexical elements σε σύνολα από φράσεις (Phrases). Κάθε Phrase περιλαμβάνει ένα σύνολο από αναφορές σε αυτά τα LexicalElements που δημιουργήθηκαν κατά τη διαδικασία του Lexical Lookup. Μια αναπαράσταση της διαδικασίας βλέπουμε στο Σχήμα 5.10.

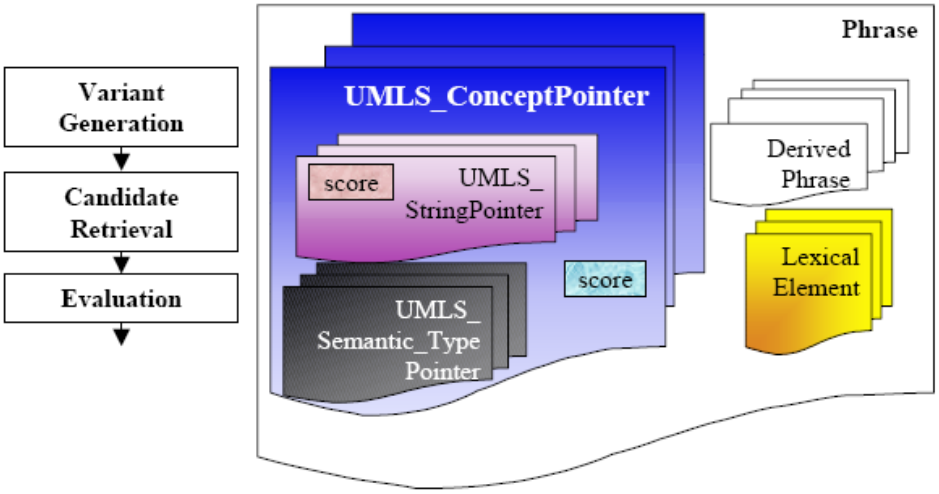
Η Container Class Derived Phrase



Σχήμα 5.11 Αναπαράσταση της Κλάσης Derived Phrase σε Σχέση με τη Δομή του Κειμένου. [73]

Η διαδικασία Variant Generator προσθέτει derived phrases στην phrase. Οι derived phrases αποτελούνται από ποικιλομορφίες των lexical elements της phrase. Οι ποικιλομορφίες αυτές προήλθαν από τον αναδρομικό συνδυασμό των spelling variants, synonyms, derivations, acronyms, acronyms expansions, abbreviations, abbreviations expansions και inflections. Το σύνολο των derived phrases προστίθεται στην κάθε Phrase. Οι derived phrases χρησιμοποιούνται εσωτερικά στο MMTx και δεν είναι άμεσα χρήσιμες στον πρρογραμματιστή. Αναπαράσταση της διαδικασίας σε σχέση με τις κλάσεις που δημιουργούνται φαίνεται στο Σχήμα 5.11.

Οι Container Classes UMLS_ConeptPointer, UMLS_StringPointer και UMLS_SemanticTypePointer



Σχήμα 5.12 Αναπαράσταση των Κλάσεων UMLS_ConeptPointer, UMLS_StringPointer και UMLS_SemanticTypePointer σε Σχέση με τη Δομή του Κειμένου. [73]

Η phrase και οι derived phrases χρησιμοποιούνται, ώστε να γίνει αντιστοιχία με τα ευρετήρια του Metathesaurus, κατά τη διάρκεια της διαδικασίας candidate retrieval. Τα UMLS strings που επιστρέφονται τοποθετούνται στους UMLS_StringPointers. Στα UMLS_Strings κρατάτε επίσης πληροφορία για τα concepts στα οποία ανήκει το κάθε string. Επίσης συλλέγεται πληροφορία για το σημασιολογικό τύπο (semantic type) του κάθε concept. Τα UMLS_String διαδοχικά τοποθετούνται στους UMLS_ConceptPointers. Τέλος, τα UMLS_ConceptPointers προστίθενται στη φράση.

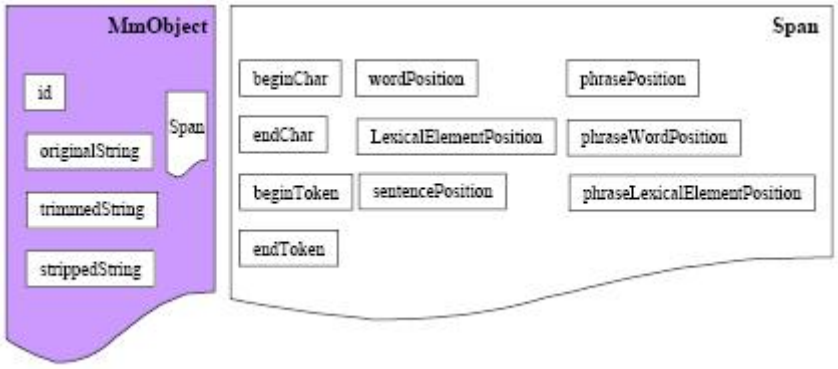
Κατά τη διάρκεια της evaluation διαδικασίας, κάθε UMLS_StringPointer αποτιμάται για το κατά πόσο καλά ταιριάζει με τη φράση. Το score που προκύπτει προστίθεται στον εκάστοτε UMLS_StringPointer. Το καλύτερο από αυτά τα string scores επέρχεται το score για τον UMLS_ConceptPointer. Το σύνολο των UMLS_ConceptPointers αποθηκεύονται ταξινομημένα κατά φθίνουσα σειρά. Στο Σχήμα 5.12 βλέπουμε μια αναπαράσταση της διαδικασίας.

Η Container Class Final Mapping

Κατά τη διάρκεια της διαδικασίας Final Mapping, γίνονται συνδυασμοί των UMLS_ConceptPointers ώστε να καλύπτουν όσο το δυνατόν καλύτερα τη φράση. Κάθε Final Mapping Περιέχει αναφορές στους UMLS_ConceptPointers που δημιουργήθηκαν κατά τη retrieval και evaluation διαδικασία. Κάθε Final Mapping αποτιμάται για το κατά πόσο καλά ο συνδυασμός των concepts καλύπτουν τη φράση. Αυτό το score προστίθεται σε κάθε Final Mapping. Το σύνολο των final mappings προστίθενται στην κάθε φράση.

Υπάρχουν εφαρμογές όπου υπάρχει ανάγκη για τις πιο κοντινές αντιστοιχίσεις από concepts και όχι για τα final mappings. Σε τέτοιες περιπτώσεις το σύνολο των UMLS_ConceptPointers της φράσης είναι πιο κατάλληλο. Το σύνολο αυτό είναι ένα ταξινομημένο κατά φθίνουσα σειρά σύνολο από concepts σύμφωνα με το evaluation score τους.

Οι Classes MmObject και Span



Σχήμα 5.13 Αναπαράσταση των Κλάσεων MmObject και Span σε Σχέση με τη Δομή του Κειμένου [73]

Υπάρχουν δύο κλάσεις, η MmObject και Span, που χρησιμοποιούνται πάρα πολύ σε NLP Tools και στο MMTx (Σχήμα 5.13).

Η MmObject κλάση κληρονομείται από σχεδόν όλες τις κλάσεις που περιγράφηκαν σε προηγούμενες ενότητες, από την Document έως την tokens. Τα instances της MmObject περιλαμβάνουν ένα Id, το πρωτότυπο sting και μια span κλάση. Τα MmObjects περιέχουν επίσης ποικιλομορφίες του πρωτότυπου string, περιλαμβανομένων δύο εκδόσεων του sting, ενός χωρίς κενά, και ενός χωρίς σημεία στίξης.

Ένα Span Object περιέχει σημεία του κειμένου, ώστε να κρατάτε αυτό το object δεμένο με το πρωτότυπο κείμενο. Ένα Span object κρατά αρχικούς και τελικούς χαρακτήρες και λέξεις σχετικά με το κείμενο και σχετικά με τη φράση.

5.2. Η Γλώσσα Οντολογιών OWL

Η OWL (Web Ontology Language) [74] είναι μια γλώσσα που χρησιμοποιείται για τον ορισμό και την υλοποίηση οντολογιών. Αναπτύχθηκε από την W3C (World Wide Web Consortium). Μια OWL οντολογία συμπεριλαμβάνει τις περιγραφές των κλάσεων, των σχέσεων και των στιγμιότυπων (classes, properties, instances). Δοθέντος μιας τέτοιας οντολογίας, μέσω της σημασιολογίας της OWL μπορούν να προκύψουν λογικά συμπεράσματα, όπως γεγονότα τα οποία δεν περιγράφονται ρητά στην οντολογία, αλλά περιέχονται στη σημασιολογία της.

Μια ερώτηση που πάντα γεννάται είναι η εξής: Τι προσφέρει αυτή η γλώσσα περισσότερο, που δεν το προσφέρει η XML?

Από την W3C δίνεται η εξής απάντηση:

- Μια OWL οντολογία διαφέρει από την XML στο ότι είναι μια αναπαράσταση της γνώσης, και όχι απλά μια τυποποίηση μηνυμάτων. Στην XML δίνεται μια λειτουργική σημασιολογία, για παράδειγμα «Αν δοθεί μια παραγγελία “PurchaseOrder, μετέφερε το ποσό “Amount” δολάρια, από τον λογαριασμό “AccountFrom” στον λογαριασμό “AccountTo” και στείλε το προϊόν “Product”». Οι εμπλεκόμενοι που χρησιμοποιούν την τυποποίηση αυτή δεν μπορούν να υποστηρίξουν συμπερασμούς εκτός από ότι έχει ορισθεί να κατανοούν.
- Ένα πλεονέκτημα των OWL οντολογιών προσδοκάται, πως είναι η δημιουργία εργαλείων που μπορούν να συμπεράνουν γνώση μέσω αυτών (reasoners). Τα εργαλεία αυτά θα παρέχουν γενική υποστήριξη που δε θα είναι συγκεκριμενοποιημένη σε ένα συγκεκριμένο πεδίο, κάτι που θα ήταν σίγουρο για ένα σύστημα που θα προσπαθούσε να παράγει συμπερασμό για μια συγκεκριμένο XML

τυποποίηση. Η δημιουργία ενός τέτοιου χρήσιμου εργαλείου συμπερασμού είναι μια πολύ επίπονη εργασία. Το χτίσιμο οντολογιών είναι πολύ πιο βολικό. Η προσδοκία της W3C είναι πως πολλές ομάδες θα εμπλακούν στην δημιουργία οντολογιών. Οι ομάδες αυτές θα επωφεληθούν από εργαλεία τρίτων, που θα βασίζονται στις ιδιότητες της OWL, εργαλεία που θα δίνουν μια ποικιλία δυνατοτήτων, που οι περισσότεροι οργανισμοί θα ήταν πολύ δύσκολο να αναπαράγουν.

5.2.1. Τα είδη της OWL

Η γλώσσα OWL παρέχει τρείς υπογλώσσες, που η μια είναι πιο εκφραστική από την άλλη.

- *OWL Lite:* Υποστηρίζει τους χρήστες που έχουν την ανάγκη για απλές ιεραρχίες και απλούς περιορισμούς. Για παράδειγμα αν και η OWL Lite υποστηρίζει τον περιορισμό πληθικότητας (cardinality constraints), επιτρέπει μόνο τις τιμές 0 και 1 ως τιμές της. Είναι πολύ ευκολότερο να παρασχεθούν εργαλεία υποστήριξης για την OWL Lite σε σύγκριση με τις πιο εκφραστικές εκδοχές της OWL.

- *OWL DL:* Υποστηρίζει τους χρήστες που χρειάζονται τη μέγιστη εκφραστικότητα χωρίς όμως να χάνουν σε υπολογιστική πληρότητα από το σύστημα συμπερασμού (εγγυάται ο υπολογισμός όλων των συνεπαγωγών που θέλει να αποδώσει η οντολογία). Η OWL DL περιλαμβάνει όλα τα δομικά συστατικά της OWL με κάποιους περιορισμούς, όπως ο διαχωρισμός στους τύπους (μια κλάση δεν μπορεί να είναι ταυτόχρονα και στιγμιότυπο, ούτε σχέση, μια σχέση δεν μπορεί ταυτόχρον ανα είναι και στιγμιότυπο, ούτε κλάση). Η OWL DL είναι τόσο διαδεδομένη λόγο της ιδιαίτερης ομοιότητάς της με την περιγραφική λογική (description logic), ενός πεδίου έχει μελετηθεί ιδιαίτερα. Η OWL DL σχεδιάστηκε να υποστηρίζει το υπαρκτό κομμάτι της περιγραφικής λογικής και έχει επιθυμητές υπολογιστικές ιδιότητες για συστήματα συνμπερασμού.

- *Η OWL Full:* Υποστηρίζει χρήστες που επιθυμούν τη μέγιστη εκφραστικότητα αλλά χωρείς εγγυήσεις σε υπολογιστική πληρότητα. Για παράδειγμα στην OWL Full μια κλάση μπορεί να τη χειριστεί κάποιος και σαν μια συλλογή από στιγμιότυπα, αλλά και

σαν στιγμιότυπο. Μια ακόμη πολύ σημαντική διαφορά από την OWL DL είναι πως ένα owl:DatatypeProperty μπορεί να τεθεί και ως owl:InverseFunctionalProperty. Η OWL Full επιτρέπει σε μια οντολογία να αυξήσει τη σημασία που αποδίδει το ορισμένο (RDF ή OWL) λεξικό. Είναι απίθανο κάποιο λογισμικό συμπερασμού να μπορέσει να υποστηρίξει κάθε χαρακτηριστικό της OWL Full.

Καθεμιά από αυτές τις υπογλώσσες είναι επέκταση της προηγούμενης απλούστερης σε εκφραστικότητα. Έτσι ισχύουν τα εξής:

- Κάθε OWL Lite οντολογία είναι και OWL DL.
- Κάθε OWL DL οντολογία είναι και OWL Full.
- Κάθε ορθό OWL Lite συμπέρασμα είναι και ορθό OWL DL συμπέρασμα.
- Κάθε ορθό OWL DL συμπέρασμα είναι και ορθό OWL Full συμπέρασμα.

Όποιος θέλει να αναπτύξει μια οντολογία κάνοντας χρήση της OWL πρέπει να λάβει υπόψη ποιο είδος ταιριάζει καλύτερα στις ανάγκες του. Στο αν θα επιλέξει OWL Lite ή OWL DL εξαρτάται από την έκταση στην οποία ο χρήστης χρειάζεται τους πιο εκφραστικούς περιορισμούς που παρέχει η OWL DL. Τα συστήματα συμπερασμού για την OWL Lite έχουν την επιθυμητή υπολογιστική ικανότητα. Η επιλογή μεταξύ OWL DL και OWL Full εξαρτάται κυρίως στην έκταση κατά την οποία οι χρήστες χρειάζονται τις ιδιότητες του meta-modeling, όπως τον ορισμό κλάσεων από κλάσεις. Όταν χρησιμοποιείται η OWL Full, η υποστήριξη στον συμπερασμό είναι λιγότερο ικανοποιητική.

5.2.2. Η Σύνταξη της OWL

Η OWL είναι επέκταση της RDF, έτσι ένα OWL έγγραφο χρησιμοποιεί σύνταξη βασισμένη στις RDF/XML και ξεκινάει πάντα με την ετικέτα <rdf:RDF….>.

5.2.2.1. Name Spaces

Αρχικά δηλώνονται κάποια απαραίτητα namespaces:

```
<rdf:RDF
  xmlns:owl ="HTTP://WWW.W3.ORG/2002/07/OWL#"
  xmlns:rdf ="HTTP://WWW.W3.ORG/1999/02/22-RDF-SYNTAX-NS#"
  xmlns:rdfs="HTTP://WWW.W3.ORG/2000/01/RDF-SCHEMA#"
  xmlns:xsd ="HTTP://WWW.W3.ORG/2001/XLMSCHEMA#">
```

5.2.2.2. Πληροφορίες για την Οντολογία

Μία OWL οντολογία μπορεί να ξεκινάει με μια συλλογή από δηλώσεις που δίνουν διάφορα στοιχεία για την οντολογία, όπως κάποια σχόλια για αυτή και προηγούμενες εκδόσεις της μέσω της <owl:Ontolgy…> ετικέτας:

```
<owl:Ontology rdf:about="">
  <rdfs:comment>An example OWL ontology
  </rdfs:comment>
  <owl:priorVersion
                rdf:resource="HTTP://WWW.MYDOMAIN.ORG/UNI-NS-OLD"/>
  <owl:imports
                rdf:resource="HTTP://WWW.MYDOMAIN.ORG/PERSONS"/>
  <rdfs:label>University Ontology</rdfs:label>
</owl:Ontology>
```

Ενώ τα namespaces χρησιμοποιούνται για λόγους αποσαφήνισης, οι imported οντολογίες παρέχουν ορισμούς που έχουν ορισθεί σε άλλες οντολογίες και που μπορούν να χρησιμοποιηθούν στην αναπτυσσόμενη.

Στο σημείο αυτό μπορεί να γίνει και χρήση των εξής ετικετών:

- owl:priorVersion: Δίνει προηγούμενες εκδόσεις της παρούσας οντολογίας.
- owl:versionInfo: Δίνει πληροφορίες για την παρούσα οντολογία.

- owl:backwardCompatibleWith: Παρέχει αναφορές σε άλλες οντολογίες. Όλα τα περιεχόμενα των προηγούμενων εκδόσεων που αναφέρονται σε αυτό το σημείο έχουν την ίδια ερμηνεία και στην παρούσα έκδοση.
- owl:incompatibleWith: Δίνει κάποιες οντολογίες με τις οποίες, η παρούσα οντολογία, πλέον δεν είναι backward compatible.

5.2.2.3. Ορισμός Κλάσεων

Οι Classes ορίζονται με την ετικέτα <owl:Class…>:

```
<owl:Class rdf:ID="associateProfessor">
  <rdfs:subClassOf
        rdf:resource="#academicStaffMember"/>
</owl:Class>
```

5.2.2.4. Διακριτές κλάσεις

Όταν είναι απαραίτητο να ορισθεί πως δύο κλάσεις περιλαμβάνουν instances που δεν μπορούν να ανήκουν και στις δύο κλάσεις, τότε χρησιμοποιείται η ετικέτα <owl:disjointWith…>:

```
<owl:Class rdf:about="#associateProfessor">
  <owl:disjointWith rdf:resource="#professor"/>
  <owl:disjointWith
        rdf:resource="#assistantProfessor"/>
</owl:Class>
```

5.2.2.5. Ισοδύναμες Κλάσεις

Για τον ορισμό κλάσεων που είναι ισοδύναμες χρησιμοποιείται η ετικέτα: <owl:equivalentClass...>:

```
<owl:Class rdf:ID="faculty">
  <owl:equivalentClass rdf:resource="#academicStaffMember"/>
</owl:Class>
```

5.2.2.6. Οι Κλάσεις Thing και Nothing

Η owl:Thing είναι η πιο γενική κλάση, που εμπεριέχει τα πάντα, ενώ η owl:Nothing είναι η άδεια κλάση. Κάθε κλάση είναι subclass της owl:Thing και superclass της owl:Nothing.

5.2.2.7. Data Type Properties και Object Properties

Στην OWL υπάρχουν δύο ειδών properties. Τα Object properties, που συσχετίζουν αντικείμενα μεταξύ τους. Παράδειγμα τέτοιου property είναι το is-TaughtBy. Το δεύτερο είδος είναι τα Data type properties, που συσχετίζουν αντικείμενα με datatype τιμές. Παραδείγματα από data type properties είναι τα “phone number”, “title”, “age” κ.α. Η OWL κάνει χρήση των data types του XML Schema π.χ.:

```
<owl:DatatypeProperty rdf:ID="age">
  <rdfs:range  rdf:resource=“&xsd;nonNegativeInteger”/>
</owl:DatatypeProperty>
```

Παρακάτω δίνονται κάποια παραδείγματα από Object Properties:

```
<owl:ObjectProperty rdf:ID="isTaughtBy">
```

```
  <owl:domain rdf:resource="#course"/>
  <owl:range  rdf:resource="#academicStaffMember"/>
  <rdfs:subPropertyOf rdf:resource="#involves"/>
</owl:ObjectProperty>
```

5.2.2.8. Inverse Properties

Όταν δύο Object Properties εκφράζουν το ένα το αντίστροφο του άλλου, τότε χρησιμοποιείται η ετικέτα <owl:inverseOf…>:

```
<owl:ObjectProperty rdf:ID="teaches">
  <rdfs:rangerdf:resource="#course"/>
  <rdfs:domain  rdf:resource="#academicStaffMember"/>
  <owl:inverseOf rdf:resource="#isTaughtBy"/>
</owl:ObjectProperty>
```

Στην παραπάνω περίπτωση, τα domain και range θα μπορούσαν να παραληφθούν, μιας που μπορούν να κληρονομηθούν από το inverse property (ουσιαστικά είναι τα ίδια με το object property “isTaughBy” αντιστρέφοντας το domain με το range).

5.2.2.9. Equivalent Properties

Για τη δήλωση δύο ισοδύναμων properties γίνεται χρήση του <owl:equivalentProperty…>:

```
<owl:ObjectProperty rdf:ID="lecturesIn">
  <owl:equivalentProperty   rdf:resource="#teaches"/>
</owl:ObjectProperty>
```

5.2.2.10. Restrictions

Στην OWL μπορεί να δηλωθεί πως μια κλάση C ικανοποιεί συγκεκριμένα κριτήρια. Αυτό σημαίνει πως όλα τα instances της κλάσης αυτής ικανοποιούν τα κριτήρια αυτά.

Αυτό είναι ισοδύναμο με το να πούμε πως η κλάση C είναι subclass της κλάσης C', που η κλάση C' είναι ένα σύνολο αντικειμένων που ικανοποιούν τα κριτήρια αυτά. Η κλάση C' μπορεί να είναι ανώνυμη (anonymous class).

Για να γίνει η παραπάνω δήλωση, χρησιμοποιούνται οι περιορισμοί, με την ετικέτα <owl:Restriction…>. Η ετικέτα αυτή περιλαμβάνει μια ακόμη ετικέτα, την <owl:onProperty…> και μια ή περισσότερες δηλώσεις από περιορισμούς.

- Ένας τύπος περιορισμού δηλώνει περιορισμούς στην πληθικότητα (at least 1, at most 3, κ.α.).
- Ένας άλλος τύπος περιορισμού ορίζει τα είδη των τιμών που κάποιο property μπορεί να λάβει.
    - owl:allValuesFrom: Δηλώνει τον καθολικό προσδιοριστή.
    - owl:hasValue: Θέτει μια συγκεκριμένη τιμή στο property.
    - owl:someValuesFrom: Δηλώνει τον υπαρξιακό προσδιοριστή.

Παράδειγμα owl:allValuesFrom περιορισμού:

```
<owl:Class rdf:about="#firstYearCourse">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#isTaughtBy"/>
      <owl:allValuesFrom rdf:resource="#Professor"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Η παραπάνω δήλωση προσδιορίζει πως τα μαθήματα πρώτου έτους διδάσκονται από professors μόνο.

Παράδειγμα owl:hasValue περιορισμού:

```
<owl:Class rdf:about="#mathCourse">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#isTaughtBy"/>
      <owl:hasValue rdf:resource="#949352"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Με τον παραπάνω περιορισμό δηλώνεται πως το μάθημα των μαθηματικών διδάσκεται από έναν συγκεκριμένο καθηγητή με ID 949352.

Παράδειγμα owl:someValuesFrom περιορισμού:

```
<owl:Class rdf:about="#academicStaffMember">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#teaches"/>
      <owl:someValuesFrom rdf:resource="#undergraduateCourse"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Που δηλώνει πως όλα τα academic staff members πρέπει να διδάσκουν τουλάχιστον ένα προπτυχιακό μάθημα.

Παράδειγμα περιορισμού στην πληθικότητα:

```
<owl:Class rdf:about="#department">
  <rdfs:subClassOf>
    <owl:Restriction>
```

```
        <owl:onProperty rdf:resource="#hasMember"/>
        <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">
        10 </owl:minCardinality>
        <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">
        30 </owl:maxCardinality>
      </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Το παραπάνω παράδειγμα υποδηλώνει πως ένα τμήμα πρέπει να έχει το λιγότερο 10 και το περισσότερο 30 μέλη.

Ένας περιορισμός, όπως προαναφέρθηκε είναι μια ανώνυμη κλάση, δεν έχει κάποιο ID, δεν ορίζεται ως owl:Class και έχει μόνο τοπική ισχύ αφού μπορεί να χρησιμοποιηθεί μόνο εκεί που εμφανίζεται ο περιορισμός. Επίσης οι κλάσεις, σύμφωνα με τα παραπάνω, έχουν δύο τρόπους εμφάνισης, κλάσεις που ορίζονται ως owl:Class με κάποιο ID, και τοπικές ανώνυμες κλάσεις που ορίζονται ως κάποιο σύνολο από αντικείμενα που ικανοποιούν κάποια συνθήκη περιορισμού ή ως συνδυασμός από άλλες κλάσεις.

5.2.2.11. Τύποι Εμφάνισης των Properties

Μέχρι στιγμής είδαμε τα properties να έχουν τον τύπο owl:inverseOf. Εκτός από αυτή τη μορφή, ένα property Μπορεί να έχει και τους εξής τύπους εμφάνισης:

- owl:TransitiveProperty: Υποδηλώνει μεταβατικότητα, π.χ. “has better grade than”, ή “is ancestor of”.
- owl:SymmetricProperty: Υποδηλώνει συμμετρία, π.χ. “has same grade as”, ή “is sibling of”.
- owl:FunctionalProperty: Ορίζει ένα property που έχει το πολύ μια τιμή για κάθε αντικείμενο στο οποίο εφαρμόζεται. Π.χ. “age”, “height”.
- owl:InverseFunctionalProperty: Ορίζει ένα property για οποίο δυο διαφορετικά αντικείμενα δεν μπορούν να έχουν την ίδια τιμή.

Παράδειγμα εφαρμογής των owl:TransitiveProperty και owl:SymmetricProperty:

```
<owl:ObjectProperty rdf:ID="hasSameGradeAs">
   <rdf:type rdf:resource="&owl;TransitiveProperty"/>
   <rdf:type rdf:resource="&owl;SymmetricProperty"/>
   <rdfs:domain rdf:resource="#student"/>
   <rdfs:range rdf:resource="#student"/>
</owl:ObjectProperty>
```

5.2.2.12. Annotation Properties

Εκτός των Data Type και Object Properties που προαναφέρθηκαν, η OWL έχει και έναν άλλο τύπο properties που ονομάζονται Annotation Properties. Τα properties αυτά χρησιμοποιούνται για να δοθούν διάφορες επισημειώσεις σε classes είτε σε instances μιας οντολογίας. Η OWL Full δεν θέτει κανέναν περιορισμό στα annotations μιας οντολογίας. Η OWL DL επιτρέπει annotation σε classes, properties, individuals αλλά μόνο υπό τις εξής συνθήκες:

- Τα σύνολα των object properties, datatype properties, annotation properties και ontology properties, πρέπει να είναι ξεχωριστά μεταξύ τους. Έτσι στην OWL ΔΛ το dc:creator δεν μπορεί να είναι συγχρόνως datatype property και annotation property.
- Τα annotation properties πρέπει ναι έχουν μια ρητή δήλωση της μορφής: AnnotationPropertyID rdf:type owl:AnnotationProperty.
- Τα annotation properties δεν πρέπει να χρησιμοποιούνται σε αξιώματα. Αυτό σημαίνει πως στην OWL DL κάποιος δεν μπορεί να ορίσει subproperties ή domain/range περιορισμούς σε annotation properties.
- Το αντικείμενο σε ένα annotation property μπορεί να είναι είτε μια τιμή (data literal), είτε μια αναφορά σε URI, είτε ένα individual.

Υπάρχουν πέντε προκαθορισμένα annotation properties από την ίδια την OWL:

- owl:versionInfo
- rdfs:label
- rdfs:comment
- rdfs:seeAlso
- rdfs:isDefined

Παράδειγμα χρήσης annotation property:

```
<owl:AnnotationProperty rdf:about="&dc;creator"/>

<owl:Class rdf:about="#MusicalWork">
        <rdfs:label>Musical work</rdfs:label>
        <dc:creator>N.N.</dc:creator>
</owl:Class>
```

5.2.2.13. Union, Intersection και Complement

Στην OWL υπάρχει και ο συνδυασμός κλάσεων με τη χρήση των Boolean operations (Union, Intersection και Complement). Παρακάτω δίνονται μερικά παραδείγματα:

```
<owl:Class rdf:about="#course">
        <rdfs:subClassOf>
            <owl:complementOf rdf:resource="#staffMember"/>
        </rdfs:subClassOf>
</owl:Class>
```

Με το πιο πάνω παράδειγμα υποδηλώνεται πως τα μαθήματα και τα μέλη του προσωπικού είναι διαφορετικά μεταξύ τους (disjoint).

```
<owl:Class rdf:ID="peopleAtUni">
        <owl:unionOf rdf:parseType="Collection">
           <owl:Class rdf:about="#staffMember"/>
```

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

Στο παραπάνω παράδειγμα η νέα κλάση δεν είναι υποκλάση της κλάσης που δημιουργείτε με την ένωση, αλλά είναι ισοδύναμή της.

```
owl:Class rdf:ID="facultyInCS">
    <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#faculty"/>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#belongsTo"/>
                <owl:hasValue rdf:resource="#CSDepartment"/>
         </owl:Restriction>
    /owl:intersectionOf>
</owl:Class>
```

Με το παραπάνω παράδειγμα υποδηλώνονται ποιο είναι το προσωπικό στο τμήμα πληροφορικής.

5.2.2.14. Enumeration

Ένα ακόμη συστατικό της OWL είναι η απαρίθμηση (enumeration), που δηλώνει τα αντικείμενα που περιέχει μια κλάση και ορίζεται με την ετικέτα <owl:oneOf…>:

```
<owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Monday"/>
    <owl:Thing rdf:about="#Tuesday"/>
    <owl:Thing rdf:about="#Wednesday"/>
    <owl:Thing rdf:about="#Thursday"/>
    <owl:Thing rdf:about="#Friday"/>
    <owl:Thing rdf:about="#Saturday"/>
```

```
        <owl:Thing rdf:about="#Sunday"/>
</owl:oneOf>
```

5.2.2.15. Δήλωση των Individuals

Η δήλωση των instances γίνεται με τον εξής τρόπο:

```
<rdf:Description rdf:ID="949352">
        <rdf:type rdf:resource="#academicStaffMember"/>
</rdf:Description>
```

Επίσης μπορεί να γίνει και με την εξής διαφοροποίηση:

```
<academicStaffMember rdf:ID="949352">
        <uni:age rdf:datatype="&xsd;integer"> 39</uni:age>
</academicStaffMember>
```

5.2.2.16. No Unique-Names Assumptions

Η OWL δεν υιοθετεί τη μοναδικότητα στις ονομασίες (No Unique-Names Assumption). Αν δύο instances έχουν διαφορετικά ονόματα ή ID αυτό δε σημαίνει πως πρόκειται για διαφορετικά instances. Για παράδειγμα αν υποθέσουμε στον τομέα των μαθημάτων, πως κάθε μάθημα έχει δηλωθεί πως διδάσκεται το πολύ από ένα μέλος του προσωπικού, και πως ένα συγκεκριμένο μάθημα διδάσκεται από δυο μέλη του προσωπικού, τότε ένας OWL reasoner δε θα αποδώσει κάποιο σφάλμα, αλλά θα συμπεράνει πως οι δύο resources των μελών του προσωπικού είναι ισοδύναμες.

Για να δηλωθεί πως δύο individuals πρέπει να αναγνωρίζονται και ως διαφορετικά, πρέπει να δηλωθεί ρητά η διαφοροποίησή τους με την ετικέτα <owl:differentFrom…>:

```
<lecturer rdf:about="#949318">
        <owl:differentFrom rdf:resource="#949352"/>
</lecturer>
```

Η OWL παρέχει και έναν τρόπο για τη δήλωση πως ανά δύο τα instances μιας λίστας είναι διαφορετικά μεταξύ τους:

```
<owl:allDifferent>
        <owl:distinctMembers rdf:parseType="Collection">
            <lecturer rdf:about="#949318"/>
            <lecturer rdf:about="#949352"/>
            <lecturer rdf:about="#949111"/>
        </owl:distinctMembers>
</owl:allDifferent>
```

5.2.2.17. Τύποι Δεδομένων

Το XML Schema παρέχει ένα μηχανισμό για τον ορισμό τύπων δεδομένων από τους ίδιους τους χρήστες, π.χ. ο τύπος δεδομένων "adultAge" περιλαμβάνει όλους ακεραίους που είναι μεγαλύτεροι από 18. Τέτοιου είδους τύποι δεδομένων δεν επιτρέπονται να χρησιμοποιηθούν στα OWL έγγραφα.

Στο έγγραφο ορισμού της OWL, του W3C, δίνεται μια λίστα όλων των τύπων δεδομένων του XML Schema που μπορούν να χρησιμοποιηθούν στην OWL. Η λίστα αυτή συμπεριλαμβάνει τους πιο συχνά χρησιμοποιήσιμους τύπους όπως string, integer, Boolean, time και date.

5.2.3. Τα Δομικά Στοιχεία της OWL Αναλόγως της Υπογλώσσας

5.2.3.1. OWL Full

Στην OWL Full όλα τα δομικά στοιχεία της Owl μπορούν να χρησιμοποιηθούν, με οποιοδήποτε συνδυασμό τους, αρκεί το αποτέλεσμα να είναι έγκυρο RDF έγγραφο.

5.2.3.2. OWL DL

Στην OWL DL κάθε resource επιτρέπεται να είναι class, data type, data type property, object property, individual, data value ή μέρος του λεξικού στο οποίο βασίζεται η OWL και δεν επιτρέπεται κανένας συνδυασμός τους. Π.χ. μια class δεν μπορεί να είναι ταυτόχρονα και individual.

Ένα Property δεν μπορεί να έχει κάποιες τιμές από data type και κάποιες από class. Αυτό θα σήμαινε πως θα ήταν συγχρόνως και data type property και object property.

Αν μια δήλωση της OWL DL κάνει χρήση κάποιων resources, οι resources που χρησιμοποιούνται θα πρέπει να είναι ορισμένες σε κάποιο σημείο του εγγράφου. Για παράδειγμα αν έχουμε την δήλωση:

```
<owl:Class rdf:ID="C1">
        <rdfs:subClassOf rdf:about="#C2"/>
</owl:Class>
```

Θα πρέπει σε κάποιο σημείο του OWL DL εγγράφου να υπάρχει και η εξής δήλωση:

```
<owl:Class rdf:ID="C2"/>
```

Στην OWL DL δεν μπορεί να ορισθεί για data type property τίποτα από τα παρακάτω:

- owl:inverseOf
- owl:FunctionalProperty
- owl:InverseFunctionalProperty
- owl:SymmetricProperty

Επίσης, δεν επιτρέπεται σε κάποιο transitive property να ορισθούν περιορισμοί στην πληθικότητα (cardinality restrictions).

Οι ανώνυμες κλάσεις που προέρχονται από περιορισμούς (restricted anonymous classes), επιτρέπεται να ορισθούν ως το domain ή το range σε ορισμό τύπου owl:equivalentClass ή owl:disjointWith, και ως το range (αλλά όχι το domain) σε δήλωση rdfs:subClassOf.

5.2.3.3. OWL Lite

Εδώ ισχύουν οι περιορισμοί της OWL DL και κάποιοι επιπλέον.

Δεν επιτρέπονται τα εξής:

- owl:one of
- owl:disjointWith
- owl:unionOf
- owl:complementOf
- owl:hasValue

Οι περιορισμοί πληθικότητας (minimal, maximal, exact) μπορούν μόνο να γίνουν στις τιμές 0 και 1.

Η owl:equivalentClass δεν μπορεί πλέον να γίνει μεταξύ ανώνυμων κλάσεων, παρά μόνο μεταξύ κλάσεων που έχουν οριστεί με κάποιο ID.

5.3. Jena OWL API

Το Jena [75] είναι μια υποδομή που μπορεί να χρησιμοποιηθεί για την κατασκευή εφαρμογών βασισμένων στη σημασιολογία.

Παρέχει διαπροσωπίες και κλάσεις για τη δημιουργία και χειρισμό οντολογιών που είναι εκφρασμένες σε OWL. Μοντελοποιεί μέσω κλάσεων μια οντολογία στη μνήμη του υπολογιστή, ώστε κάποιος να είναι σε θέση να την εκμεταλλευτεί και να τη χρησιμοποιήσει σε διάφορες εφαρμογές.

5.3.1. Δομικά Στοιχεία του Jena OWL API

Η OWL όπως έχει προαναφερθεί είναι μια επέκταση της RDF. Αυτό εκφράζεται και μέσω του Jena, αφού οι σχετικές με OWL classes και interfaces επεκτείνουν ή χρησιμοποιούν της classes και interfaces του RDF API του Jena.

5.3.1.1. .OntModel

Είναι η αφετηρία χρήσης του Jena. Δημιουργεί το μοντέλο της οντολογίας στη μνήμη του υπολογιστή για περεταίρω εκμετάλλευσή της. Περιλαμβάνει όλες τις δηλώσεις που εκφράζονται στην οντολογία και χρησιμοποιείται για την ανάκτηση υπαρχόντων resources (classes, individuals, properties κ.α.) ή και για τη δημιουργία νέων.

5.3.1.2. OntClass

Μέσω του OntClass δημιουργούνται αντικείμενα που αναπαριστούν κλάσεις της οντολογίας. Οι μέθοδοι που συμπεριλαμβάνει μπορούν να χρησιμοποιηθούν για την εμφάνιση, λήψη και χρήση των instances, superclasses, subclasses, restrictions κ.α. μιας συγκεκριμένης κλάσης.

Οι classes μπορεί να είναι είτε απλά ονόματα κάτω από τα οποία κατηγοριοποιείται ένα πλήθος από instances, αλλά μπορεί να είναι και πιο πολύπλοκες δομές, οι οποίες περιγράφονται με τη χρήση ορισμών άλλων κλάσεων, μέσω των union, intersection, complement, enumeration και restrictions. Το OntModel παρέχει μεθόδους για τη δημιουργία τέτοιων πολύπλοκων κλάσεων.

5.3.1.3. OntProperty

Τα properties αναπαρίστανται στο jena μέσω της OntProperty διαπροσωπίας. Η OntProperty παρέχει μεθόδους για τον ορισμό των domain και range ενός property, όπως επίσης τον τύπο του property (DatatypeProperty, ObjectProperty, SymmetricProperty, FunctionalProperty κ.α.). Επίσης μπορούν να ορισθούν σχέσεις subproperty και superproperty. Τα properties ορίζονται χωρίς να έχουν εξάρτηση από μια συγκεκριμένη κλάση.

5.3.2. Παραδείγματα Χρήσης του Jena API

```
//Δημιουργία ενός κενού μοντέλου οντολογίας
OntModel m = ModelFactory.createOntologyModel();
String ns = new String("http://www.example.com/onto1#");
String baseURI = new String("http://www.example.com/onto1");
Ontology onto = m.createOntology(baseURI);

// Δημιουργία των κλάσεων 'Person', 'MalePerson' and 'FemalePerson'
OntClass person = m.createClass(ns + "Person");
OntClass malePerson = m.createClass(ns + "MalePerson");
OntClass femalePerson = m.createClass(ns + "FemalePerson");

// Οι FemalePerson και MalePerson είναι subclasses of Person
person.addSubClass(malePerson);
person.addSubClass(femalePerson);

// Οι FemalePerson και MalePerson είναι disjoint
malePerson.addDisjointWith(femalePerson);
femalePerson.addDisjointWith(malePerson);

// Δημιουργεία του datatype property 'hasAge'
DatatypeProperty hasAge = m.createDatatypeProperty(ns + "hasAge");

// Το 'hasAge' δέχεται ακέραιες τιμές, άρα η range του είναι 'integer'
// Οι βασικοί τύποι δεδομένων ορίζονται στο πακέτο 'vocabulary'
hasAge.setDomain(person);
hasAge.setRange(XSD.integer); // com.hp.hpl.jena.vocabulary.XSD

// Δημιουργία κάποιον individuals
Individual john = malePerson.createIndividual(ns + "John");
Individual jane = femalePerson.createIndividual(ns + "Jane");
Individual bob = malePerson.createIndividual(ns + "Bob");
```

```
// Δημιουργία της δήλωσης 'John hasAge 20'
Literal age20 =m.createTypedLiteral("20", XSDDatatype.XSDint);
Statement johnIs20 =m.createStatement(john, hasAge, age20);
m.add(johnIs20);

// Δημιουργία του object property 'hasSibling'
ObjectProperty hasSibling = m.createObjectProperty(ns + "hasSibling");

// Ορισμός των Domain και Range του property 'hasSibling' είναι η κλάση 'Person'
hasSibling.setDomain(person);
hasSibling.setRange(person);

// Δημιουργία της δήλωσης 'John hasSibling Jane‘
// και της δήλωσης 'Jane hasSibling John'
Statement siblings1 = m.createStatement(john, hasSibling, jane);
Statement siblings2 = m.createStatement(jane, hasSibling, john);

//Δημιουργία ενός κενού μοντέλου οντολογίας
OntModel m = ModelFactory.createOntologyModel();
String ns = new String(“http://www.example.com/onto1#”);
String baseURI = new String(“http://www.example.com/onto1”);
Ontology onto = m.createOntology(baseURI);

// Δημιουργία των κλάσεων ‘Person’, ‘MalePerson’ and ‘FemalePerson’
OntClass person = m.createClass(ns + "Person");
OntClass malePerson = m.createClass(ns + "MalePerson");
OntClass femalePerson = m.createClass(ns + "FemalePerson");

// Οι FemalePerson και MalePerson είναι subclasses of Person
person.addSubClass(malePerson);
person.addSubClass(femalePerson);

// Οι FemalePerson και MalePerson είναι disjoint
```

```
malePerson.addDisjointWith(femalePerson);
femalePerson.addDisjointWith(malePerson);

// Δημιουργεία του datatype property 'hasAge'
DatatypeProperty hasAge = m.createDatatypeProperty(ns + "hasAge");

// Το 'hasAge' δέχεται ακέραιες τιμές, άρα η range του είναι 'integer'
// Οι βασικοί τύποι δεδομένων ορίζονται στο πακέτο ‘vocabulary’
hasAge.setDomain(person);
hasAge.setRange(XSD.integer); // com.hp.hpl.jena.vocabulary.XSD

// Δημιουργία κάποιον individuals
Individual john = malePerson.createIndividual(ns + "John");
Individual jane = femalePerson.createIndividual(ns + "Jane");
Individual bob = malePerson.createIndividual(ns + "Bob");

// Δημιουργία της δήλωσης 'John hasAge 20'
Literal age20 =m.createTypedLiteral("20", XSDDatatype.XSDint);
Statement johnIs20 =m.createStatement(john, hasAge, age20);
m.add(johnIs20);

// Δημιουργία του object property 'hasSibling'
ObjectProperty hasSibling = m.createObjectProperty(ns + "hasSibling");

// Ορισμός των Domain και Range του property 'hasSibling' είναι η κλάση 'Person'
hasSibling.setDomain(person);
hasSibling.setRange(person);

// Δημιουργία της δήλωσης 'John hasSibling Jane‘
// και της δήλωσης 'Jane hasSibling John'
Statement siblings1 = m.createStatement(john, hasSibling, jane);
Statement siblings2 = m.createStatement(jane, hasSibling, john);
```

5.4. Reasoners και Pellet

5.4.1. Οι Reasoners στην OWL

Οι reasoners [76] είναι εργαλεία που χρησιμοποιούνται για την εξαγωγή επιπρόσθετης πληροφορίας. Ουσιαστικά χρησιμοποιούν τη γνώση που έχει ορισθεί από το χρήστη, και μέσω αυτής και κατάλληλων κανόνων, εξάγουν νέα γνώση που αφήνεται να εννοείται από τα αξιώματα που έχουν δοθεί.

Δύο είναι οι βασικές υπηρεσίες που προσφέρουν οι reasoners στην OWL:

- Έλεγχο της ορθότητας της οντολογίας.
- Συμπερασμός νέας γνώσης.

Ένα απλό παράδειγμα συμπερασμού νέας γνώσης είναι το εξής:

Penguin subClassOf Bird
Pablo type Penguine

Με τις παραπάνω δηλώσεις σε μία οντολογία, μέσω ενός reasoner μπορεί πλέον να συμπεραθεί πως ο Pablo είναι τύπου Bird.

Στη συνέχεια δίνεται ένα παράδειγμα μη ορθής περιγραφής γνώσης:

Bird subClassOf FlyingThing
Penguin subClassOf Bird
Penguin disjointWith FlyingThing

Αν δοθεί η παραπάνω περιγραφή γνώσης σε έναν reasoner, τότε αυτός θα εξάγει ένα σφάλμα ορθότητας. Αυτό θα συμβεί διότι μέσω των παραπάνω δηλώσεων γίνεται ο συμπερασμός, πως οι penguins ανήκουν και στα Flying Things και στα Not Flying Things.

5.4.2. Ο Pellet Reasoner

Ο Pellet είναι ένας reasoner που χρησιμοποιείται για OWL-DL οντολογίες.

- Καλύπτει όλα τα δομικά συστατικά της OWL-DL.
- Είναι υλοποιημένος σε Java και είναι ανοιχτού κώδικα (open source).
- Καλύπτει μια μεγάλη πλειάδα από εργαλεία και συντάκτες οντολογιών, όπως το Jena, το OWL-API και το Protégé.

Η ανάπτυξή του άρχισε μέσα στο 2003, από το MINDSWAP group του πανεπιστημίου του Maryland. Προτεργάτης της δημιουργίας του ήταν ο Bijan Parsia και υλοποιήθηκε από τον Evren Sirin.

5.4.3. Χρήση του Pellet στο Jena API

Για να χρησιμοποιήσει κάποιος τον Pellet reasoner μέσω του Jena API, ώστε να μπορέσει να εξαχθεί επιπλέον πληροφορία μιας οντολογίας, πρέπει να κάνει άμεση χρήση της διαπροσωπίας που προσφέρει ο Pellet.

Παρακάτω δίνεται ένα παράδειγμα χρήσης του Pellet σε ένα μοντέλο οντολογίας του Jena:

```
// ontology that will be used
String ont = "http://www.mindswap.org/2004/owl/mindswappers";

// create an empty ontology model using Pellet spec
OntModel model =
ModelFactory.createOntologyModel( PelletReasonerFactory.THE_SPEC );

// read the file
model.read( ont );

// get the instances of a class
OntClass Person = model.getOntClass( "http://xmlns.com/foaf/0.1/Person" );
Iterator instances = Person.listInstances();
```

Στο παραπάνω παράδειγμα δίνεται σαν όρισμα ο Pellet reasoner στα specifications του μοντέλου αναπαράστασης της οντολογίας, που πάει να δημιουργηθεί από το Jena API. Στη συνέχεια διαβάζεται το αρχείο που περιέχει την οντολογία, ώστε να δημιουργηθεί στη μνήμη

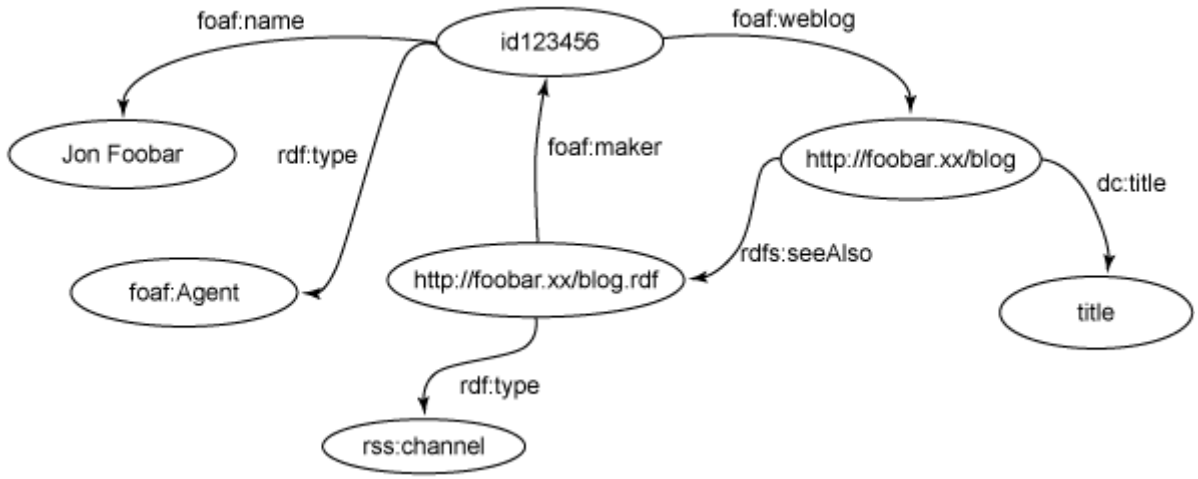
του υπολογιστή η οντολογία. Τέλος ανακτάται μια κλάση από το μοντέλο και έπειτα τα instances αυτής της κλάσης. Λόγω του ότι χρησιμοποιείται ο Pellet reasoner, πλέον στο σύνολο των instances δε θα συμπεριλαμβάνονται μόνο τα άμεσα instances που έχουν ορισθεί από τον δημιουργό της οντολογίας, αλλά και τα instances όλων των κλάσεων που έχουν δηλωθεί ως υποκλάσεις αυτής της κλάσης.

5.5. Η Γλώσσα ερωτήσεων SPARQL

Η SPARQL [77, 78] είναι η γλώσσα επερωτήσεων οντολογιών που προτείνει η W3C. Είναι βασισμένη σε προηγούμενες γλώσσες επερωτήσεων RDF γράφων όπως την rdfDB, και RDQL με επιπρόσθετα χαρακτηριστικά. Υλοποιήσεις της SPARQL υπάρχουν για μια ποικιλία από πλατφόρμες και γλώσσες, αλλά εδώ θα παρουσιαστεί η χρήση της μέσω του Jena API [75].

5.5.1. Ένα Απλό Ερώτημα SPARQL

Στο Σχήμα 5.14 δίνεται μέρος του RDF γράφου που θα χρησιμοποιηθεί στην παρουσίαση της SPARQL.



Σχήμα 5.14 Οντολογίας σε RDF. [78]

Παρακάτω δίνεται ένα απλό παράδειγμα ερωτήματος SPARQL που λέει το εξής: «Βρες το URL του blog του ατόμου με όνομα Jon Foobar».

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
        SELECT ?url
        FROM    <bloggers.rdf>
        WHERE  {
           ?contributor foaf:name "Jon Foobar" .
           ?contributor foaf:weblog ?url .
        }
```

Η πρώτη γραμμή του ερωτήματος απλά ορίζει ένα PREFIX για το FOAF namespace, αυτό γίνεται για να μην χρειάζεται να το γράφει ολόκληρο κάποιος κάθε φορά που θέλει να αναφερθεί σε ένα resource του γράφου. Η γραμμή με το SELECT ορίζει το τι πρέπει να επιστρέψει το ερώτημα, στην περίπτωση αυτή ένα url. Στην SPARQL τα ονόματα των μεταβλητών αρχίζουν με ένα ? ή $. Το FROM είναι προαιρετικό, και παρέχει το URI όπου βρίσκεται το μοντέλο που πρόκειται να ερωτηθεί. Στο παράδειγμα εδώ, δεικτοδοτείται ένα τοπικό αρχείο, αλλά θα μπορούσε να είναι και ένα URL του Web. Τέλος η γραμμή με το WHERE αποτελείται από μια σειρά από πρότυπα τριπλέτων. Οι τριπλέτες αυτές αποτελούν το επονομαζόμενο graph pattern.

Το ερώτημα προσπαθεί να κάνει ένα ταίριασμα των τριπλέτων του graph pattern με αυτές που περιλαμβάνει το μοντέλο. Κάθε ταίριασμα τριπλέτων που περιλαμβάνει και μια μεταβλητή του ερωτήματος, αποτελεί και μια λύση στο ερώτημα “query solution”, και οι τιμές των μεταβλητών που περιλαμβάνονται στην πρόταση του SELECT είναι μέρος της απάντησης του ερωτήματος.

Στο συγκεκριμένο παράδειγμα, η πρώτη τριπλέτα στην WHERE πρόταση κάνει ταίριασμα με έναν κόμβο του γράφου που έχει ένα property foaf:name με τιμή “Jon Foobar”, και δίνει την τιμή που θα βρεθεί ως αντικείμενο της τριπλέτας, στην μεταβλητή “contributor”. Στη συνέχεια αφού η μεταβλητή contributor πάρει κάποια τιμή, με τη δεύτερη γραμμή της WHERE πρότασης, γίνεται προσπάθεια να βρεθούν τριπλέτες στο μοντέλο που να έχουν σαν υποκείμενο τον contributor που βρέθηκε στο προηγούμενο βήμα, και αυτός να έχει με τη σειρά του ένα property foaf:weblog. Η τιμή αυτού του property θα αποτελεί και ένα αποτέλεσμα του ερωτήματος.

5.5.2. Εκτέλεση SPARQL Ερωτημάτων Μέσω του Jena API

Η SPARQL είναι διαθέσιμη μέσω του Jena με ένα API που ονομάζεται ARQ. Για να εκτελέσουμε ένα SPARQL ερώτημα μέσω java κώδικα, γίνεται χρήση των κλάσεων που εμπεριέχονται στο πακέτο com.hp.hp1.jena.query. Η QueryFactory συμπεριλαμβάνει πολλές create() μεθόδους για το διάβασμα ενός ερωτήματος είτε από ένα αρχείο, είτε από ένα String. Η μέθοδος αυτή επιστρέφει ένα Query Object, το οποίο εμπεριέχει το ερώτημα προς εκτέλεση.

Το επόμενο βήμα είναι η δημιουργία ενός instance της QueryExecution, η κλάση αυτή αναπαριστά μια απλή εκτέλεση ενός ερωτήματος. Για να πάρουμε ένα QueryExecution, πρέπει να γίνει κλήση της συνάρτησης QueryExecutionFactory.create(query, model), περνώντας σαν όρισμα σε αυτή το Query προς εκτέλεση και το Model της οντολογίας που έχει δημιουργηθεί από το jena. Λόγω του ότι τα δεδομένα για το ερώτημα παρέχονται προγραμματιστικά, το query δεν χρειάζεται την FROM πρόταση.

Για να εκτελεστεί ένα απλό SELECT ερώτημα, πρέπει να κληθεί η συνάρτηση execSelect(), η οποία επιστρέφει ένα ResultSet. Το ResultSet επιτρέπει στον προγραμματιστή να διασχίσει τα αποτελέσματα του QuerySolution που επιστράφηκαν από την εκτέλεση του ερωτήματος.

Παρακάτω δίνεται ένας απλός τρόπος συνδυασμού των παραπάνω βημάτων. Εκτελεί ένα ερώτημα επάνω στα δεδομένα του RDF γράφου που δόθηκε παραπάνω και εξάγει τα αποτελέσματα στην οθόνη:

```
//Άνοιγμα του RDF γράφου από τοπικό αρχείο
InputStream in = new FileInputStream(new File("bloggers.rdf"));

// Δημιουργία ενός άδειου μοντέλου στη μνήμη και εμπλουτισμός του από τον  //γράφο
Model model = ModelFactory.createMemModelMaker().createModel();
model.read(in,null); // null base URI, since model URIs are absolute
in.close();
```

```
// Δημιουργία ενός νέου ερωτήματος
String queryString =
        "PREFIX foaf: <http://xmlns.com/foaf/0.1/> " +
        "SELECT ?url " +
        "WHERE {" +
        "      ?contributor foaf:name \"Jon Foobar\" . " +
        "      ?contributor foaf:weblog ?url . " +
        "      }";

Query query = QueryFactory.create(queryString);

// Εκτέλεση του ερωτήματος και λήψη των αποτελεσμάτων
QueryExecution qe = QueryExecutionFactory.create(query, model);
ResultSet results = qe.execSelect();

// Εμφάνιση των αποτελεσμάτων
ResultSetFormatter.out(System.out, results, query);

// Απελευθέρωση των πόρων του συστήματος που χρησιμοποιήθηκαν για την
//εκτέλεση του ερωτήματος.
qe.close();
```

5.5.3. Πιο Πολύπλοκα SPARQL Ερωτήματα

Στις οντολογίες μπορεί να υπάρχουν κόμβοι που έχουν μεν ίδιο τύπο, αλλά να έχουν συμπληρωμένα διαφορετικά properties μεταξύ τους. Παρακάτω δίνεται ένας μικρός γράφος , εκφρασμένος σε Turtle σύνταξη [79]. Περιλαμβάνει περιγραφές από τέσσερα πρόσωπα, αλλά το καθένα από αυτά έχει διαφορετικό σύνολο ιδιοτήτων.

```
        @prefix foaf: <http://xmlns.com/foaf/0.1/> .

        _:a foaf:name       "Jon Foobar" ;
           foaf:mbox        <mailto:jon@foobar.xx> ;
```

```
    foaf:depiction    <http://foobar.xx/2005/04/jon.jpg> .

_:b foaf:name         "A. N. O'Ther" ;
    foaf:mbox         <mailto:a.n.other@example.net> ;
    foaf:depiction    <http://example.net/photos/an-2005.jpg> .

_:c foaf:name         "Liz Somebody" ;
    foaf:mbox_sha1sum "3f01fa9929df769aff173f57dec2fe0c2290aeea"

_:d foaf:name         "M Benn" ;
    foaf:depiction    <http://mbe.nn/pics/me.jpeg> .
```

5.5.3.1. Προαιρετικά Ταιριάσματα (Optional Matches)

Υποθέστε πως κάποιος επιθυμεί να δημιουργήσει ένα ερώτημα που θα επιστρέφει τα ονόματα κάθε προσώπου που περιγράφεται από τον παραπάνω γράφο, μαζί με μια σύνδεση σε μια φωτογραφία του, εάν είναι διαθέσιμη. Ένα SELECT ερώτημα του οποίου το graph pattern περιλαμβάνει το foaf:depiction, θα εντοπίζει τρία αποτελέσματα. Η Liz Somebody δε θα αποτελεί αποτέλεσμα του ερωτήματος, γιατί συμπεριλαμβάνει μια foaf:name ιδιότητα στην περιγραφή της, αλλά όχι μια foaf:depiction ιδιότητα, και είναι απαραίτητο να υπάρχουν και οι δύο ιδιότητες για να αποτελέσουν αποτέλεσμα στο ερώτημα.

Η SPARQL δίνει την επιλογή του OPTIONAL keyword. Με την εντολή αυτή δίνεται η επιλογή να μην απορρίπτονται ως αποτελέσματα ενός ερωτήματος, κόμβοι που δεν περιλαμβάνουν όλες τις ιδιότητες που περιγράφονται στην WHERE πρόταση.

Παρακάτω δίνεται ένα παράδειγμα χρήσης του OPTIONAL. Το ερώτημα επιστρέφει όλα τα ονόματα των προσώπων που περιγράφονται από τον γράφο, και προαιρετικά επιστρέφει το foaf:depiction, εάν αυτό υπάρχει:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?depiction
WHERE {
```

```
    ?person foaf:name ?name .
    OPTIONAL {
       ?person foaf:depiction ?depiction .
    } .
  }
```

Τα αποτελέσματα του παραπάνω ερωτήματος, δίνονται σχηματικά παρακάτω:

```
-----------------------------------------------------------------------
      | name            | depiction                                   |
      =======================================================================
      | "A. N. O'Ther"  | <http://example.net/photos/an-2005.jpg>     |
      | "Jon Foobar"    | <http://foobar.xx/2005/04/jon.jpg>          |
      | "Liz Somebody"  |                                             |
      | "M Benn"        | <http://mbe.nn/pics/me.jpeg>                |
-----------------------------------------------------------------------
```

5.5.3.2. Εναλλακτικά Ταιριάσματα (Alternative Matches)

Πολλές φορές χρησιμοποιείται το e-mail κάποιου για να αναγνωριστεί μοναδικά κάποιο πρόσωπο. Για λόγους απορρήτου, κάποιοι προτιμούν να χρησιμοποιούν κάποια συνάρτηση κατακερματισμού (hashcodes) στο e-mail τους, αντί του ίδιου του e-mail τους. Τα e-mails εκφρασμένα με το κανονικό τους κείμενο δίνονται μέσω του foaf:mbox property, ενώ οι hashcodes των e-mail μέσω του foaf:mbox_shalsum property, στον αρχικό RDF γράφο που δόθηκε στην αρχή του υποκεφαλαίου.. Τα δύο αυτά properties είναι συνήθως αμοιβαία αλληλοαποκλειόμενα. Σε τέτοιες περιπτώσεις η SPARQL δίνει την alternative matches ιδιότητα, ώστε να γραφούν ερωτήματα που θα επιστρέφουν όποιο από τα δύο properties είναι διαθέσιμο.

Οι alternative matches ορίζονται δίνοντας περισσότερα από ένα εναλλακτικά graph patterns στο ερώτημα, με το UNION keyword ανάμεσά τους. Το ερώτημα που δίνεται ως παράδειγμα πιο κάτω, επιστρέφει τα ονόματα των ατόμων που αναπαρίστανται στον RDF γράφο, μαζί με είτε το foaf:mbox είτε το foaf:mbox_shalsum που τους αντιστοιχεί:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?name ?mbox
```

```
WHERE {
 ?person foaf:name ?name .
 {
    { ?person foaf:mbox ?mbox } UNION { ?person foaf:mbox_sha1sum ?mbox }
 }
}
```

Εάν βρεθούν να συνυπάρχουν και οι δύο ιδιότητες ενός ατόμου, τότε θα εμφανιστούν και δύο διαφορετικά αποτελέσματα.

Πιο κάτω παρουσιάζονται τα αποτελέσματα του προηγούμενου ερωτήματος:

```
-----------------------------------------------------------------------
      | name                  |       mbox
      |
      ========================================================================
      | "Jon Foobar"          | <MAILTO:JON@FOOBAR.XX>                      |
      | "A. N. O'Ther"        | <mailto:a.n.other@example.net>              |
      | "Liz Somebody"        |"3f01fa9929df769aff173f57dec2fe0c2290aeea" |
-----------------------------------------------------------------------
```

5.5.3.3. Περιορισμοί στις Τιμές (Value Constraints)

Το FILTER keyword στην SPARQL περιορίζει τα αποτελέσματα ενός ερωτήματος θέτοντας περιορισμούς στις τιμές των μεταβλητών που ζητούνται να επιστραφούν. Οι περιορισμοί αυτοί είναι λογικές εκφράσεις που αποτιμούνται σε boolean τιμές, και μπορεί να συνδυαστούν με λογικούς τελεστές "&&" και ""||". Για παράδειγμα ένα ερώτημα που επιστρέφει μια λίστα από ονόματα, μπορεί να περιοριστεί με ένα φίλτρο, ώστε να επιστρέφει μόνο ονόματα που ταιριάζουν σε μια κανονική έκφραση "regular expression".

Παρακάτω δίνεται ένα παράδειγμα ερωτήματος που χρησιμοποιεί το FILTER keyword:

```
PREFIX rss:   <http://purl.org/rss/1.0/>
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
PREFIX  dc:   <http://purl.org/dc/elements/1.1/>
SELECT ?item_title ?pub_date
WHERE {
```

```
?item rss:title ?item_title .
?item dc:date ?pub_date .
FILTER xsd:dateTime(?pub_date) >= "2005-04-01T00:00:00Z"^^xsd:dateTime
      && xsd:dateTime(?pub_date) < "2005-05-01T00:00:00Z"^^xsd:dateTime
}
```

Στο παράδειγμα αυτό, περιορίζεται η ημερομηνία δημοσίευσης σε ένα συγκεκριμένο εύρος ημερομηνιών.

Για τη χρήση κανονικών εκφράσεων, χρησιμοποιείται το φίλτρο “FILTER regex(?x,’name’,’i’)”. Με το φίλτρο αυτό λέμε στο ερώτημα να ταιριάξει τη μεταβλητή x, έτσι ώστε να εμπεριέχει σε κάποιο σημείο της το κείμενο “name”, και με το όρισμα ‘i’ δηλώνουμε πως το ταίριασμα αυτό είναι case insensitive.

ΚΕΦΑΛΑΙΟ 6. ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΑΝΑΠΤΥΞΗ ΤΗΣ ΟΝΤΟΛΟΓΙΑΣ

6.1. Ανάλυση απαιτήσεων

Όπως προαναφέρθηκε, ένας από τους στόχους της μεταπτυχιακής αυτής διατριβής, είναι ο σχεδιασμός μιας οντολογίας στην καρδιολογία, και πιο συγκεκριμένα στο πεδίο των καρδιαγγειακών νοσημάτων. Στο κεφάλαιο αυτό θα αναπτύξουμε την όλη διαδικασία που ακολουθήθηκε για το σχεδιασμό της, και την υλοποίησή της.

Αφού μελετήσαμε το τι είναι μια οντολογία, κάποιες γενικές οντολογίες που έχουν ως πεδίο ενδιαφέροντος την ιατρική (κυρίως το UMLS) καθώς και κάποια εργαλεία που θα μας βοηθούσαν στη σχεδίαση και ανάπτυξή της (Protégé, MMTx), ήρθαμε σε επαφή με έναν γιατρό του πεδίου.

Ο γιατρός σύμφωνα με τη θεωρία των οντολογιών είναι για εμάς ο ειδικός του πεδίου (domain expert). Του εξηγήσαμε το τι είναι μια οντολογία καθώς και το τι θέλουμε να κάνουμε, δηλαδή μια οντολογία στην καρδιολογία και ένα σύστημα για ανάκτηση κειμένων. Όποτε κατά τη σχεδίαση τέθηκαν οι εξής απαιτήσεις:

- **Εύρεση του βασικού κορμού της οντολογίας.** Στη φάση αυτή έπρεπε ο γιατρός να μας δώσει τις βασικές έννοιες που χρησιμοποιούνται στην καρδιαγγειακή νόσο (οι έννοιες αυτές είναι σύμφωνα με τη γλώσσα των οντολογιών, οι ριζικές κλάσεις ιεραρχίας ή σύμφωνα με την ορολογία οι Root Hierarchy Classes). Αυτές οι κλάσεις

είναι οι εξής: coronary artery disease, complication, diagnosis, differential diagnosis, etiology, pathophysiology, Prognosis, risk factors, treatment. Επίσης επειδή επρόκειτο η οντολογία να χρησιμοποιηθεί για την ανάκτηση κειμένων, προσθέσαμε μια επιπλέον κλάση στον βασικό κορμό της, την document, η οποία θα έχει σαν instances, τα id των κειμένων που θα εισαχθούν στην οντολογία και πάνω στα οποία θα γίνει η αναζήτηση.

- **Εύρεση των υποκλάσεων του βασικού κορμού.** Στη φάση αυτή ο γιατρός έπρεπε να μας δώσει τις έννοιες που αποτελούν υποκατηγορίες των ριζικών κλάσεων. Η ιεραρχία της οντολογίας φαίνεται στο σχήμα 6.1.
- **Εύρεση των σχέσεων που θα υπάρξουν στην οντολογία.** Οι σχέσεις όπως ορίζονται από την OWL-DL είναι τα object properties. Τα properties αυτά ορίζονται μεταξύ instances της οντολογίας (ένα παράδειγμα είναι ότι το instance stemi που είναι ένα στιγμιότυπο της καρδιαγγειακής ασθένειας, myocardial infarction, έχει σαν μια μορφή θεραπείας το φάρμακο morphine, που είναι Instance της κλάσης pharmacotherapy που είναι υποκλάση της κλάσης therapy, αυτό θα αναπαρασταθεί στην οντολογία με μια σχέση “hastTreatment” μεταξύ του instance stemi και του instance morphine). Αφού η οντολογία μας έχει να κάνει με την καρδιοαγγειακή νόσο, η οποία είναι ριζική κλάση και έχει τις υποκλάσεις της καθώς και τα instances της, και οτιδήποτε άλλο εισάγεται στην οντολογία θα πρέπει να συσχετίζεται με την κλάση αυτή και πιο ειδικά με κάποια από τα instances της, πάρθηκε η εξής απόφαση για τα object properties τα οποία θα χρησιμοποιηθούν: δημιουργία ενός object property και του αντιστρόφου του (inverse property), για κάθε κλάση του βασικού κορμού της οντολογίας εκτός της coronary artery disease. Έτσι οι σχέσεις - object properties που θα χρησιμοποιηθούν είναι οι εξής:
    - hasComplication
    - hasDiagnosis
    - hasDifferentialDiagnosis
    - hasEtiology
    - hasPathophysiology
    - hasPrognosis
    - hasRiskFactors
    - hasTreatment
    - hasDocument

και οι αντίστροφές τους (ή κατά την ορολογία των οντολογιών "inverse properties")

- isComplicationOf
- isDiagnosisOf
- isDifferentialDiagnosisOf
- isEtiologyOf
- isPathophysiologyOf
- isPrognosisOf
- isRiskFactorsOf
- isTreatmentOf
- isDocumentOf.

Οι ορισμοί των σχέσεων αυτών, με τα domain και range τους, δίνετε στον Πίνακα 6.1.

- **Εύρεση των instances της κάθε κλάσης της οντολογίας μας.** Στη φάση αυτή ο γιατρός μας δίνει τα instances που πρέπει να αποτελέσουν τα στιγμιότυπα της εκάστοτε κλάσης της οντολογίας μας.
- **Συνώνυμα για κλάσεις και instances.** Λόγω του ότι η οντολογία προορίζετε για ανάκτηση κειμένων είναι σίγουρο πως θα χρειαστούμε και αρκετά συνώνυμα, τόσο για την κάθε κλάση όσο και για το κάθε instance, ώστε αν κάποιος κάνει μια αναζήτηση με κάποιον όρο τότε να γίνει έλεγχος στην οντολογίας μας, για το αν αυτός εμπεριέχετε είτε ως όνομα μιας κλάσης ή instance, είτε ως συνώνυμου τους.
- **Όλα τα δομικά στοιχεία της οντολογίας από το UMLS.** Λόγω του ότι θέλουμε οι όροι της οντολογίας μας να είναι όσο το δυνατόν ευρέως αποδεκτοί, πριν τους εισάγουμε στην οντολογία μας, αναζητήσαμε να τους βρούμε στο UMLS, το οποίο όπως προαναφέρθηκε εμπεριέχει τους ευρέως αποδεκτούς όρους της ιατρικής από μια πλειάδα ιατρικών λεξικών που έχουν αναπτυχθεί ανά τον κόσμο. Επίσης χρησιμοποιήθηκε το UMLS TAB του Protégé, το οποίο δίνει τη δυνατότητα να γίνει αναζήτηση κάποιου όρου απευθείας στο UMLS. Αν κάποιος όρος βρεθεί μέσω του εργαλείου αυτού, τότε μας επιτρέπει να τον εισάγουμε στην οντολογία μας είτε ως κλάση, είτε ως instance, και μαζί με τον όρο εισάγει και μια πλειάδα πληροφοριών που τον αφορούν, όπως το cui του concept που εντόπισε, τον ορισμό του, κάποια γενικότερα και κάποια ειδικότερα concepts της έννοιας που μας ενδιαφέρει, και κάποια συνώνυμά του. Βέβαια υπήρξαν και περιπτώσεις που κάποιες έννοιες που ο

γιατρός μας ήθελε να συμπεριληφθούν στην οντολογία, δεν βρέθηκαν μέσω του UMLS TAB, οπότε και τις εισάγαμε στην οντολογία σύμφωνα με την κρίση του γιατρού, και με κάποια συνώνυμα της επιλογής του.

- **Μετατροπή της οντολογίας σε OWL-DL.** Λόγω του ότι η OWL-DL είναι η πιο εκφραστική μορφή της owl για τη δημιουργία οντολογιών, και λόγω του ότι οι reasoners που χρησιμοποιούνται για να εξάγουν την επιπλέον πληροφορία που μπορεί να εξαχθεί μέσω μιας οντολογίας που είναι σε owl και υπονοείται είτε μέσω της κληρονομικότητας, είτε μέσω κάποιων περιορισμών που μπορούν να εφαρμοστούν σε κάποια properties της οντολογίας, λειτουργούν πολύ πιο αποτελεσματικά με οντολογίες που είναι εκφρασμένες σε OWL-DL, από την αρχή της όλης διαδικασίας, η σκέψη μας ήταν να εκφράσουμε και τη δική μας οντολογίας σε OWL DL. Βέβαια σύμφωνα με την προηγούμενη απαίτηση, της χρήσης του UMLS TAB του Protégé, είχαμε τον περιορισμό να δουλέψουμε αρχικά με μια οντολογία εκφρασμένη σε frames project, μιας που το UMLS TAB μπορούσε να δουλέψει μόνο με τέτοιου είδους projects του Protégé. Τα frames projects, είναι ένα είδος έκφρασης οντολογιών, που χρησιμοποιήθηκε από την ομάδα του protégé, προτού τεθεί η OWL σαν επίσημη γλώσσα οντολογιών από την W3C. Πλέον όλοι οι εμπλεκόμενοι στον τομέα των οντολογιών προσπαθούν να εκφράσουν τις οντολογίες τους στη γλώσσα OWL. Έτσι βασικό μας μέλημα ήταν αφού ολοκληρωθεί η οντολογία μας στο protégé μέσω του UMLS TAB, να τη μετατρέψουμε σε OWL-DL. Το protégé δίνει από μόνο του τη δυνατότητα να εξάγουμε την οντολογία σε OWL. Με τη διαδικασία αυτή έχουμε μεν την οντολογία μας σε OWL αλλά όχι σε OWL-DL. Οι διάφοροι validators που κυκλοφορούν, μας δίνουν πως μετά τη διαδικασία εξαγωγής της οντολογίας σε OWL μέσω του protégé, αυτή είναι πλέον εκφρασμένη σε OWL-FULL και όχι OWL-DL. Ο λόγος που συμβαίνει αυτό είναι γιατί όπως προαναφέρθηκε, μαζί με τα concepts και instances που εισήχθησαν μέσω του UMLS TAB εισήχθησαν και επιπλέον πληροφορίες όπως τα synonyms τους. Με την μετατροπή του αρχείου σε OWL, δημιουργήθηκαν στην οντολογία κάποια properties, τα οποία είναι datatype properties όπως το synonym και τα οποία εφαρμόζονται και σε Classes και σε instances. Αυτό όμως είναι κάτι το οποίο δεν μπορεί να συμβεί στην OWL-DL, μιας που σε αυτή είναι επιτρεπτό να εφαρμόζονται τα datatypes properties μόνο σε instances και τα object properties μόνο μεταξύ instances. Στην εργασία μας όμως χρειαζόμαστε τα συνώνυμα και για τις Classes και για τα instances, μιας που αν

κάποιος κάνει μια ερώτηση στο σύστημα αναζήτησης εγγράφων, μπορεί να τύχει να αναζητήσει για μια γενική κλάση ή ακόμη για ένα εξειδικευμένο instance, οπότε θα πρέπει να υπάρχουν συνώνυμα και στα δύο είδη δομικών στοιχείων της οντολογίας μας. Το μόνο δομικό στοιχείο της OWL-DL που μας επιτρέπει κάτι τέτοιο, είναι τα annotation properties. Αυτά επιτρέπεται να χρησιμοποιηθούν και σε κλάσεις και σε instances ταυτόχρονα. Οπότε πρέπει να μετατρέψουμε τα datatype properties που εισήχθησαν από το UMLS TAB σε annotation properties. Επίσης το Protégé εισάγει μια κλάση KB_ROOT και οποιαδήποτε άλλη κλάση δημιουργήσαμε, τη θέτει σαν τύπου KB_ROOT. Οπότε εμείς αφαιρούμε τον ορισμό αυτής της κλάσης, και ορίζουμε τις δικές μας κλάσεις να είναι τύπου OWL με το κατάλληλο OWL tag <owl:Class>. Με τις μετατροπές αυτές έχουμε πλέον το αρχείο της οντολογίας μας σε μορφή OWL-DL.

Πίνακας 6.1 Ορισμός των Object Properties

Τίτλος σχέσης	**Domain**	**Range**
hasComplication	Coronary artery disease	Complication
hasDiagnosis	Coronary artery disease	Diagnosis
hasDifferentialDiagnosis	Coronary artery disease	Differential Diagnosis
hasEtiology	Coronary artery disease	Etiology
hasPathophysiology	Coronary artery disease	Pathophysiology
hasPrognosis	Coronary artery disease	Prognosis
hasRiskFactors	Coronary artery disease	Risk Factors
hasTreatment	Coronary artery disease	Treatment
hasDocument	Coronary artery disease	Document
isComplicationOf	Complication	Coronary artery disease
isDiagnosisOf	Diagnosis	Coronary artery disease
isDifferentialDiagnosisOf	Differential Diagnosis	Coronary artery disease
isEtiologyOf	Etiology	Coronary artery disease
isPathophysiologyOf	Pathophysiology	Coronary artery disease
isPrognosisOf	Prognosis	Coronary artery disease

isRiskFactorsOf	Risk Factors	Coronary artery disease
isTreatmentOf	Treatment	Coronary artery disease
isDocumentOf	Document	Coronary artery disease

6.2. Το Protégé στην Ανάπτυξη της Οντολογίας

Το protégé [80], είναι ένα από τα πιο γνωστά και πολυχρησιμοποιημένα εργαλεία για δημιουργία και υποστήριξη οντολογιών. Έχει αναπτυχθεί από το Stanford University, είναι γραμμένο σε java, ανοιχτό λογισμικό, και έχει μια πλειάδα από επιπρόσθετα προγράμματα (plugins) για την υποστήριξη του χτισίματος μιας οντολογίας. Είναι ένα εργαλείο που υποστηρίζει το χτίσιμο οντολογιών και με τη frame τεχνολογία, αλλά και με OWL.[81]

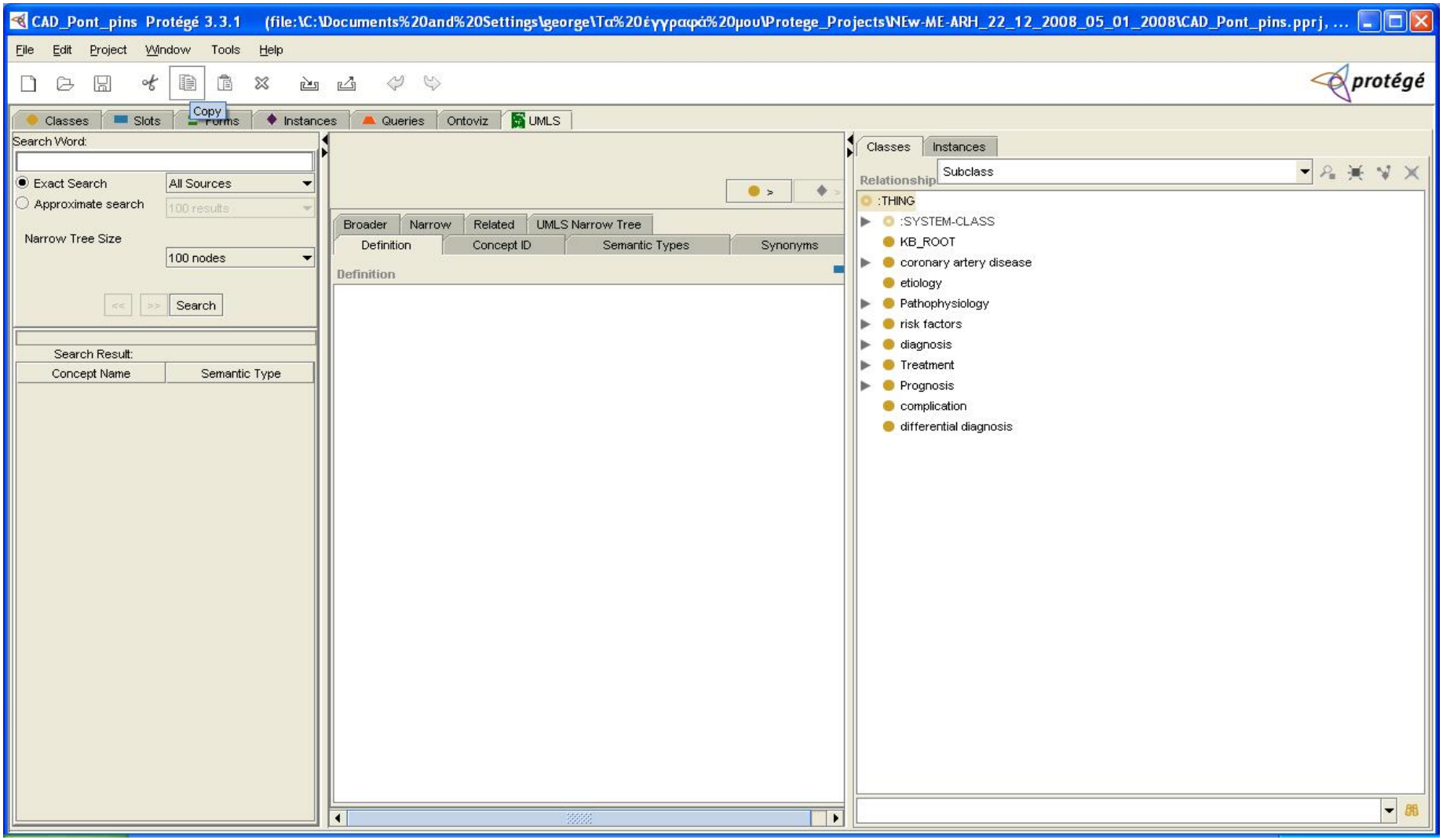
Στην παρουσίαση που ακολουθεί παρουσιάζουμε τα βασικά μέρη του protégé, σύμφωνα με τον τρόπο που χρησιμοποιήθηκε στην εργασία μας.

Όπως αναφέρεται στις απαιτήσεις της ανάπτυξης της οντολογίας μας, ήταν ανάκγη να γίνει χρήση του UMLS για να πάρουμε από αυτό concepts που είναι ευρέος αποδεκτά στην ιατρική κοινότητα, καθώς και συνώνυμα που συμπεριλαμβάνονται σε αυτά. Επίσης, θέλαμε να κάνουμε χρήση της OWL για την αναπαράσταση της γνώσης στο πεδίο, λόγω του ότι είναι η γλώσσα οντολογιών που τείνει να γίνεται η ευρέως αποδεκτή από τη διεθνή κοινότητα, και πιο συγκεκριμένα της OWL-DL για να μπορεί να γίνει αποτελεσματική χρήση κάποιου reasoner για την εξαγωγή επιπρόσθετης πληροφορίας.

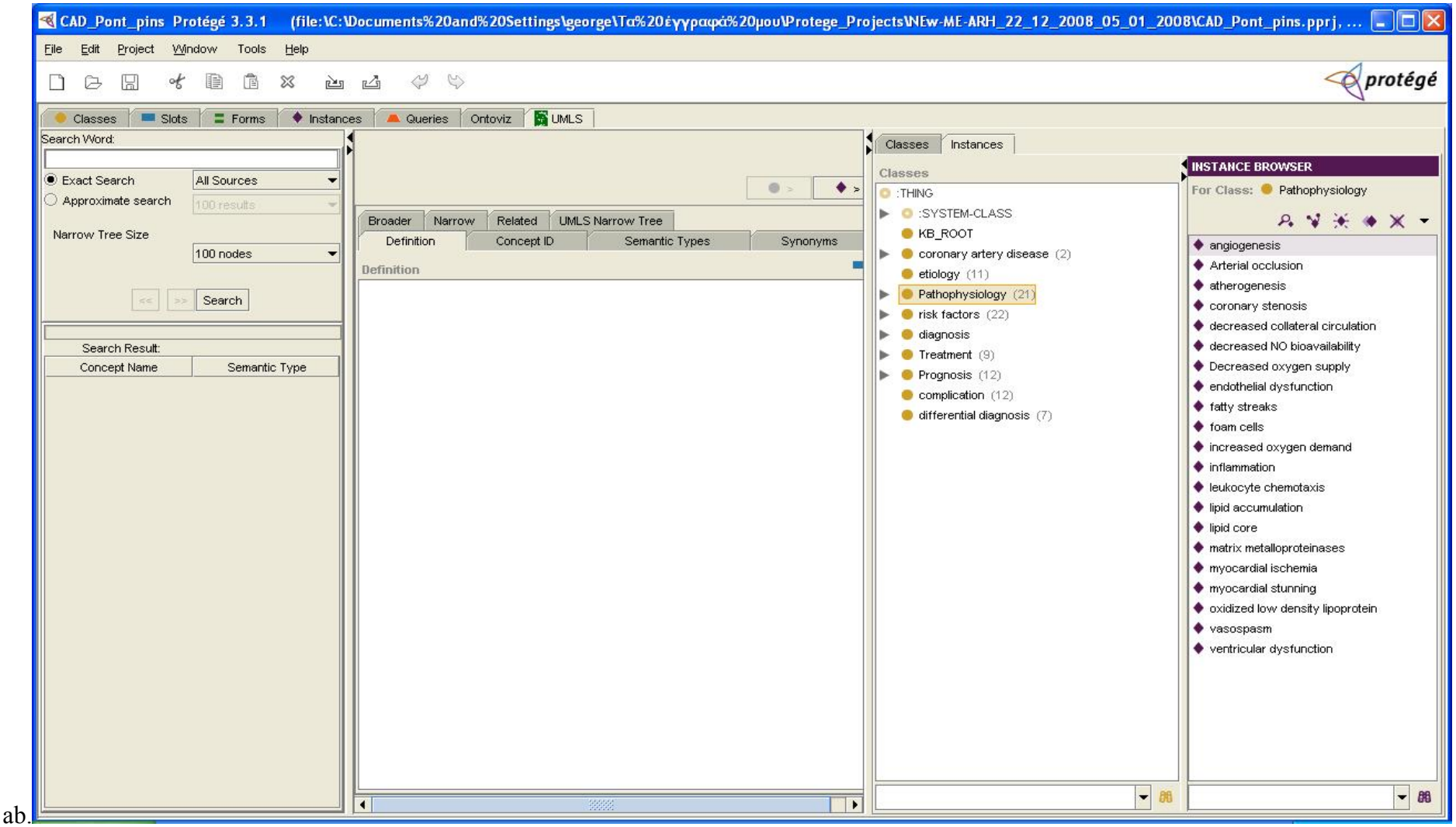
Το Protégé υποστηρίζεται από ένα plugin που ονομάζεται UMLS TAB. Το εργαλείο αυτό βοηθάει στην εύρεση concepts από το UMLS και στην εισαγωγή τους απείθειας σε μια οντολογία, με όλη την πληροφορία που τα περιβάλουν, είτε ως classes είτε ως instances. Το plugin αυτό όμως υποστηρίζεται από projects του protégé που είναι βασισμένα σε frames. Έτσι έπρεπε να δουλέψουμε αρχικά με τέτοιου είδους project, μέχρι να βρούμε όλη την πληροφορία από το UMLS, που θέλαμε να εισάγουμε στην οντολογία μας, και έπειτα να μεταφέρουμε την οντολογίας στην πιο εκφραστική γλώσσα OWL-DL.

6.2.1. Χρήση του UMLS Tab του Protégé

Στα Σχήματα 6.1 και 6.2 βλέπουμε το UMLS Tab του protégé. Όπως φαίνεται δίνει τη δυνατότητα να εισάγουμε κάποιον όρο και να κάνουμε αναζήτηση εάν αυτός ο όρος υπάρχει στο UMLS. Εάν υπάρχει, επιστρέφει τα concepts του UMLS που ταιριάζουν με τον όρο, μαζί και με επιπρόσθετη πληροφορία για αυτά τα concepts.



Σχήμα 6.1 Εισαγωγή Κλάσεων Μέσω του UMLS

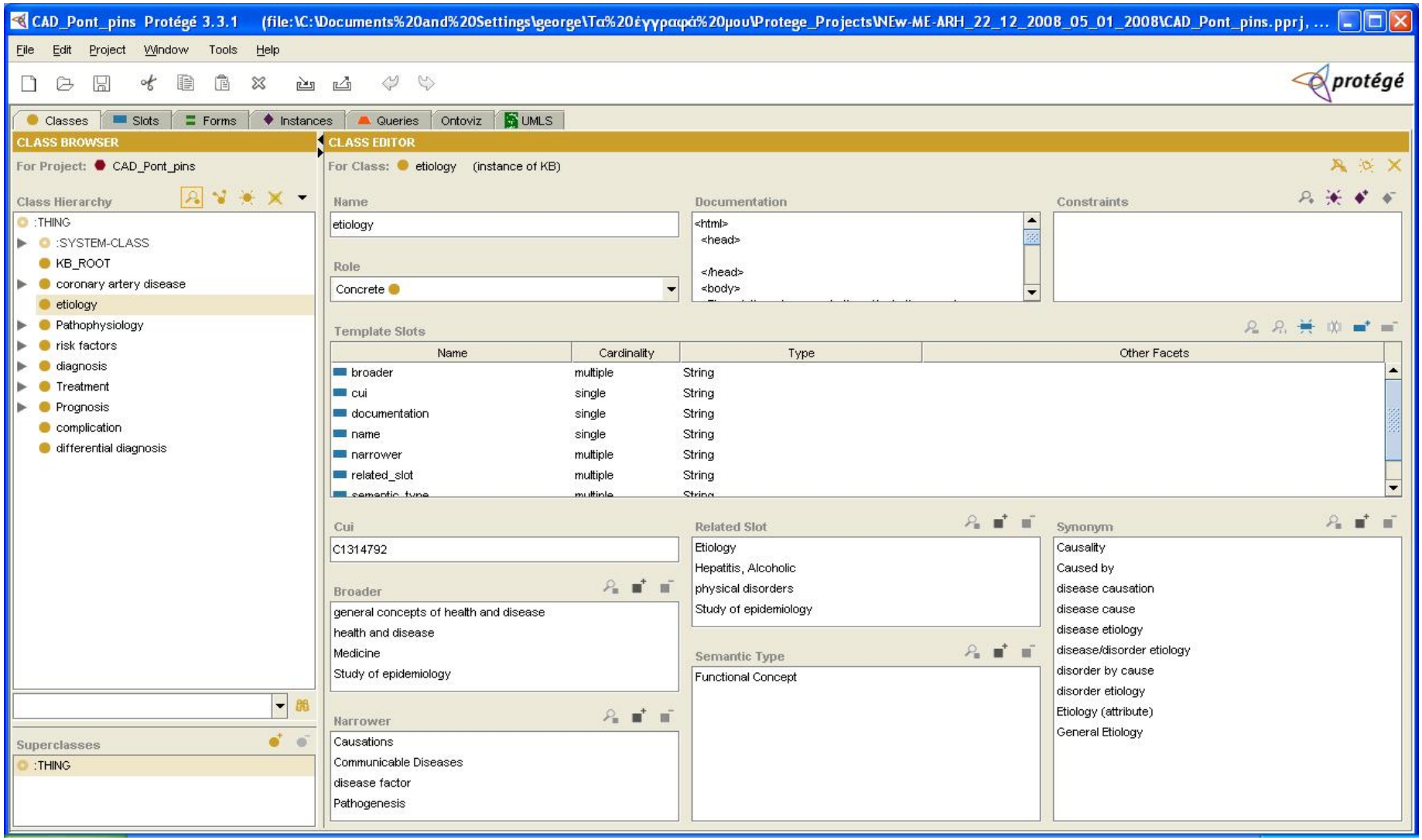


ab.

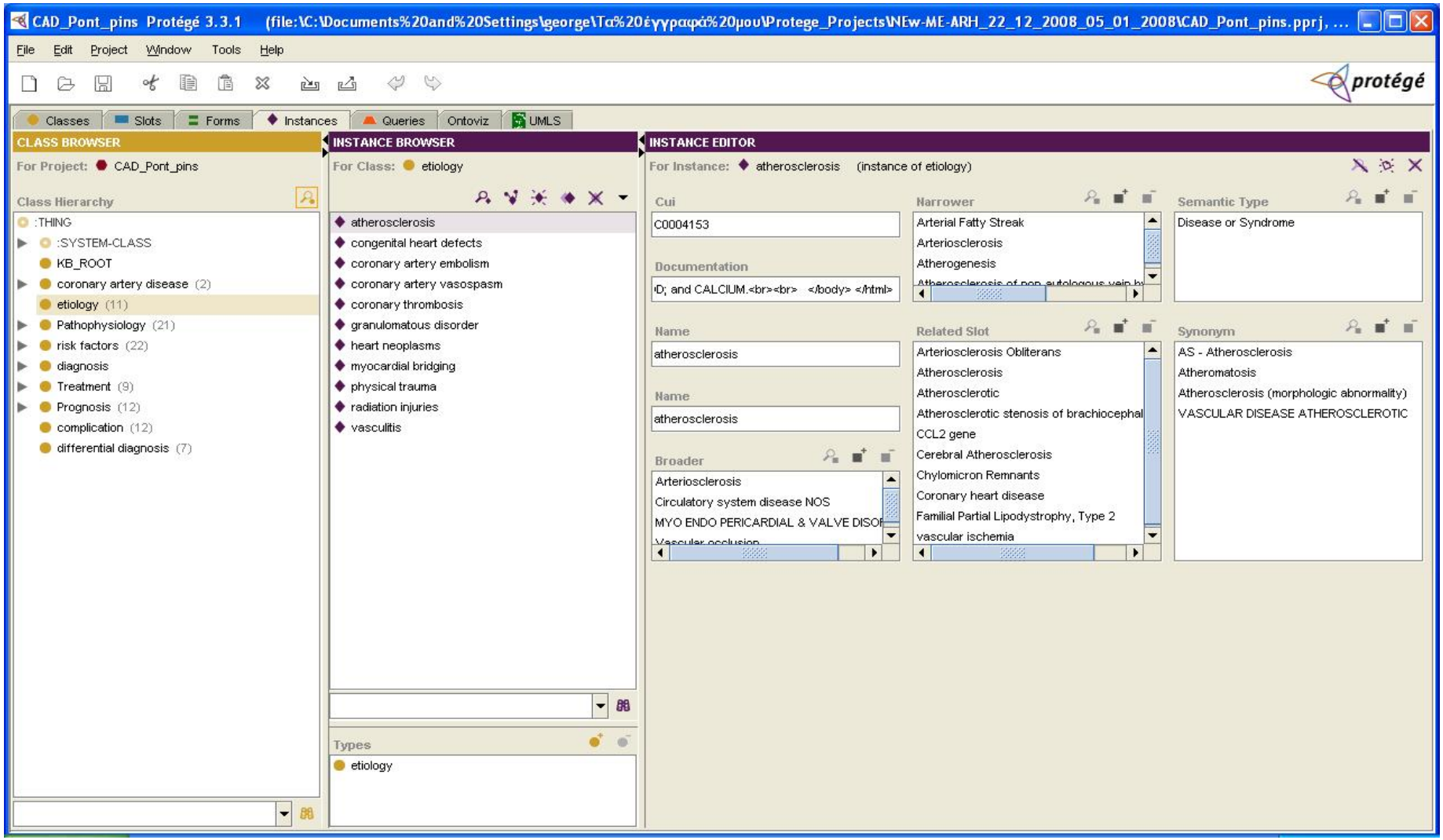
Σχήμα 6.2 Εισαγωγή Instances Μέσω του UMLS Tab.

Στο Σχήμα 6.1 είναι επιλεγμένη η καρτέλα Classes (επάνω δεξιά), που μας επιτρέπει να ορίσουμε ένα concept του UMLS ως κλάση της οντολογίας μας. Στο Σχήμα 6.2 είναι επιλεγμένη η καρτέλα instances που μας δίνει την επιλογή να εισάγουμε ένα concept του UMLS ως instance στην οντολογία μας.

Αφού εισάγουμε κάποια classes ή instances στην οντολογία, πηγαίνοντας στις αντίστοιχες καρτέλες Classes και Instances του protégé, μπορούμε να δούμε την πληροφορία που εισήχθηκε. Οι αντίστοιχες καρτέλες φαίνονται στα Σχήματα 6.3 και 6.4 αντίστοιχα.



Σχήμα 6.3 Η Καρτέλα Classes του Protege.



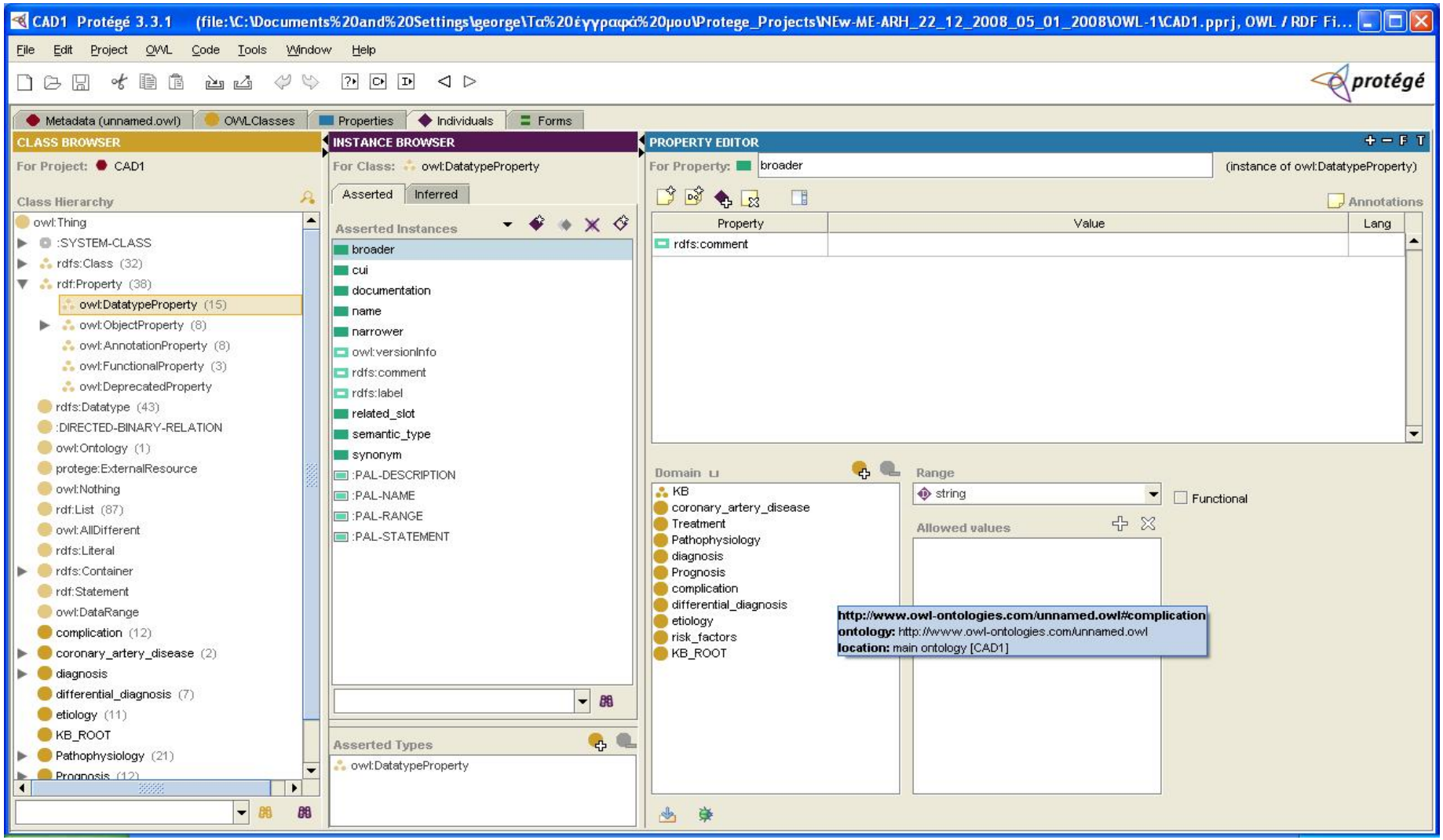
Σχήμα 6.4 Η Καρτέλα Individuals του UMLS.

6.2.2. Μετατροπή της οντολογίας σε OWL DL

Το protege δίνει την επιλογή να εξάγουμε την οντολογίας μας σε OWL. Με μια τέτοια όμως μετατροπή, εισάγει στο OWL αρχείο, εκτός των concepts και instances που χρειαζόμαστε για την αναπαράσταση του πεδίου των καρδιαγγειακών ασθενειών και κάποιους επιπρόσθετους ορισμούς των κλάσεων KB και KB_ROOT. Τις κλάσεις αυτές τις χρειάζεται το protégé για να αναπαραστήσει την πληροφορία που εισέρχεται από το UMLS. Έτσι επεμβαίνουμε χειροκίνητα στο αρχείο που εξάχθηκε και διαγράφουμε τους ορισμούς αυτών των κλάσεων καθώς και αναφορές σε αυτούς (π.χ. κάθε κλάση ορίζεται σαν τύπου KB κλάσης, οπότε εμείς αλλάζουμε τέτοιου είδους ορισμούς, ώστε η κάθε κλάση της OWL οντολογίας μας να είναι τύπου owl:Class και όχι KB class).

Επόμενο βήμα είναι η αλλαγή όλων των datatype properties που εισήχθησαν στην οντολογία, λόγο της επιπρόσθετης πληροφορίας για το κάθε concept του UMLS, όπως τα συνώνυμα (synonym property), σε annotation properties. Αυτή η μετατροπή γίνεται γιατί στην OWL και OWL DL δεν επιτρέπεται ο ορισμός ενός datatype property πάνω σε κλάση. Σύμφωνα όμως με τις απαιτήσεις ανάπτυξης της οντολογίας μας, είναι απαραίτητο να κρατάμε συνώνυμα των ονομάτων των κλάσεων μας, και αυτό μπορεί να επιτευχθεί σε OWL DL μόνο με τα annotation properties).

Αφού ανοίξουμε το OWL αρχείο με το protégé, πηγαίνουμε από το μενού Project, στο Configuration και έπειτα στην καρτέλα Options. Από εκεί ενεργοποιούμε την επιλογή “Display Hidden Frames”, μετακινούμαστε στην καρτέλα “Individuals” και επιλέγουμε από αριστερά τα “Datatype Properties”. Στο σημείο αυτό μετακινούμε με drug and drop τα datatype properties που εισήλθαν αυτόματα από το UMLS Tab στην κατηγορία “Annotation Properties”. Το τελευταίο αυτό στάδιο φαίνεται στο Σχήμα 6.5.



Σχήμα 6.5 Η Καρτέλα Individuals κατά την μετατροπή Datatype Properties σε Annotation Properties.

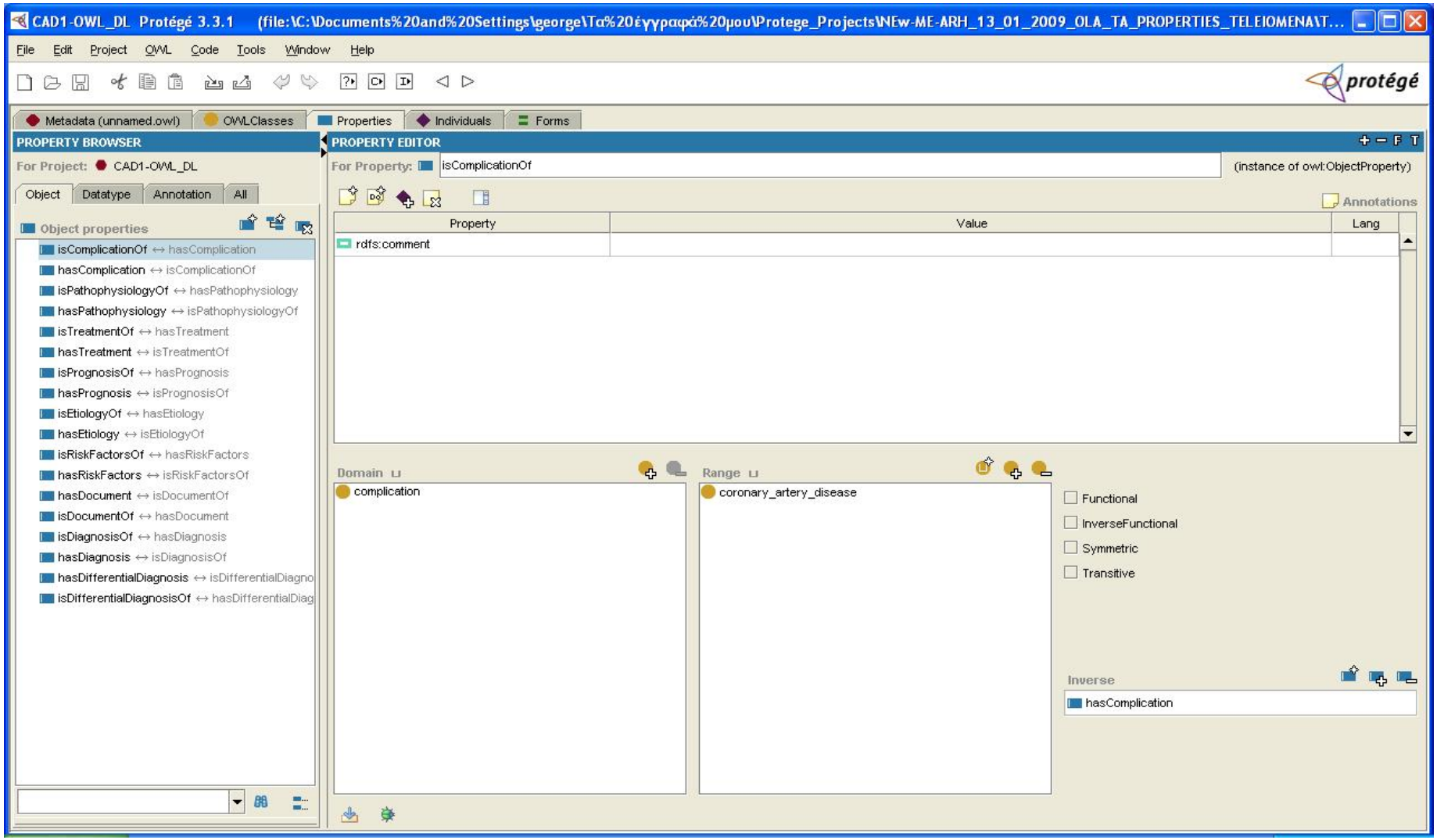
Τέλος διαγράφουμε χειροκίνητα και πάλι τους ορισμούς των “Domains” και “Ranges” από τους ορισμούς του εκάστοτε annotation property, μιας που σύμφωνα με την OWL DL ένα annotation property δεν επιτρέπεται να έχει τέτοιου είδους ορισμούς.

Για να ελέγξουμε ότι η οντολογία μας συνάδει πλέον με την OWL DL γλώσσα οντολογιών την εισάγουμε σε έναν OWL DL validator όπως τον [82].

6.2.3. Ορισμός των Σχέσεων της Οντολογίας

Πλέον περνάμε στη φάση ορισμού των object properties της οντολογίας μας. Στη φάση αυτή πρέπει να ορίσουμε μέσω του protégé, τα object properties του Πίνακα 6.1. Αυτό γίνεται από την καρτέλα Properties του protégé που φαίνεται στο Σχήμα 6.6.

Στο σημείο αυτό μπορούμε να δημιουργήσουμε τα object properties. Ορίζουμε τα domains και ranges τους, καθώς επίσης μπορούμε να ορίσουμε αν ένα property είναι Functional, InverseFunctional, Symmetric, Transitive ή το Inverse κάποιου άλλου.

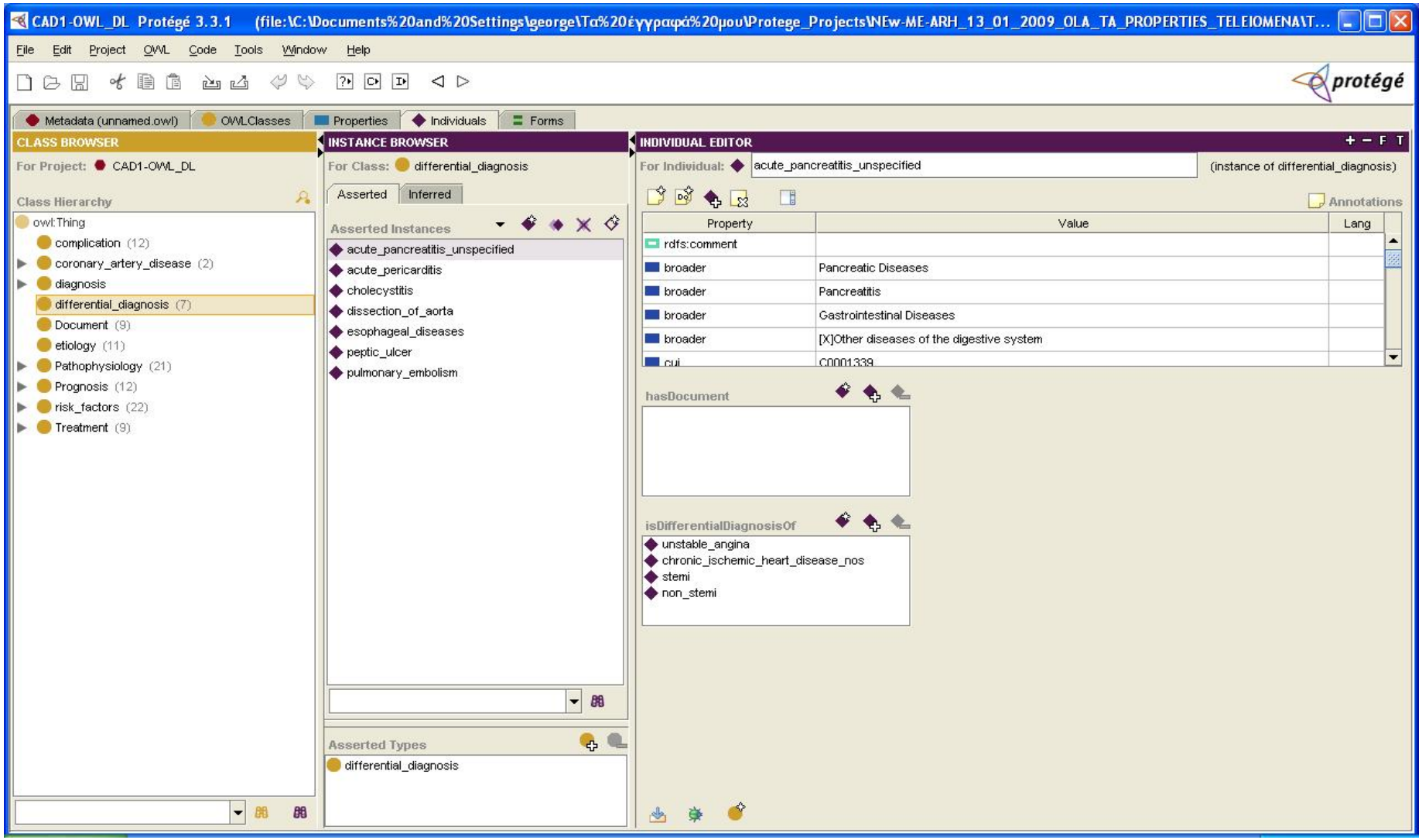


Σχήμα 6.6 . Η Καρτέλα Properties του Protégé.

6.2.4. Ορισμός Σχέσεων Μεταξύ Individuals

Στο σημείο αυτό είμαστε έτοιμοι να θέσουμε σχέσεις μεταξύ συγκεκριμένων Individuals. Αυτό γίνεται από την καρτέλα individuals του protégé (Σχήμα 6.7).

Στην καρτέλα αυτή επιτρέπεται πατώντας επάνω σε ένα individual μιας κλάσης, και επιλέγοντας την κατάλληλη σχέση που έχει ως domain την κλάση αυτή, να επιλέξουμε με πια individuals των κλάσεων που έχουν ορισθεί ως range του property αυτού, συσχετίζεται.



Σχήμα 6.7 Η Καρτέλα Individuals κατά τον Ορισμό Σχέσεων Μεταξύ Instances.

6.3. Η Οντολογία

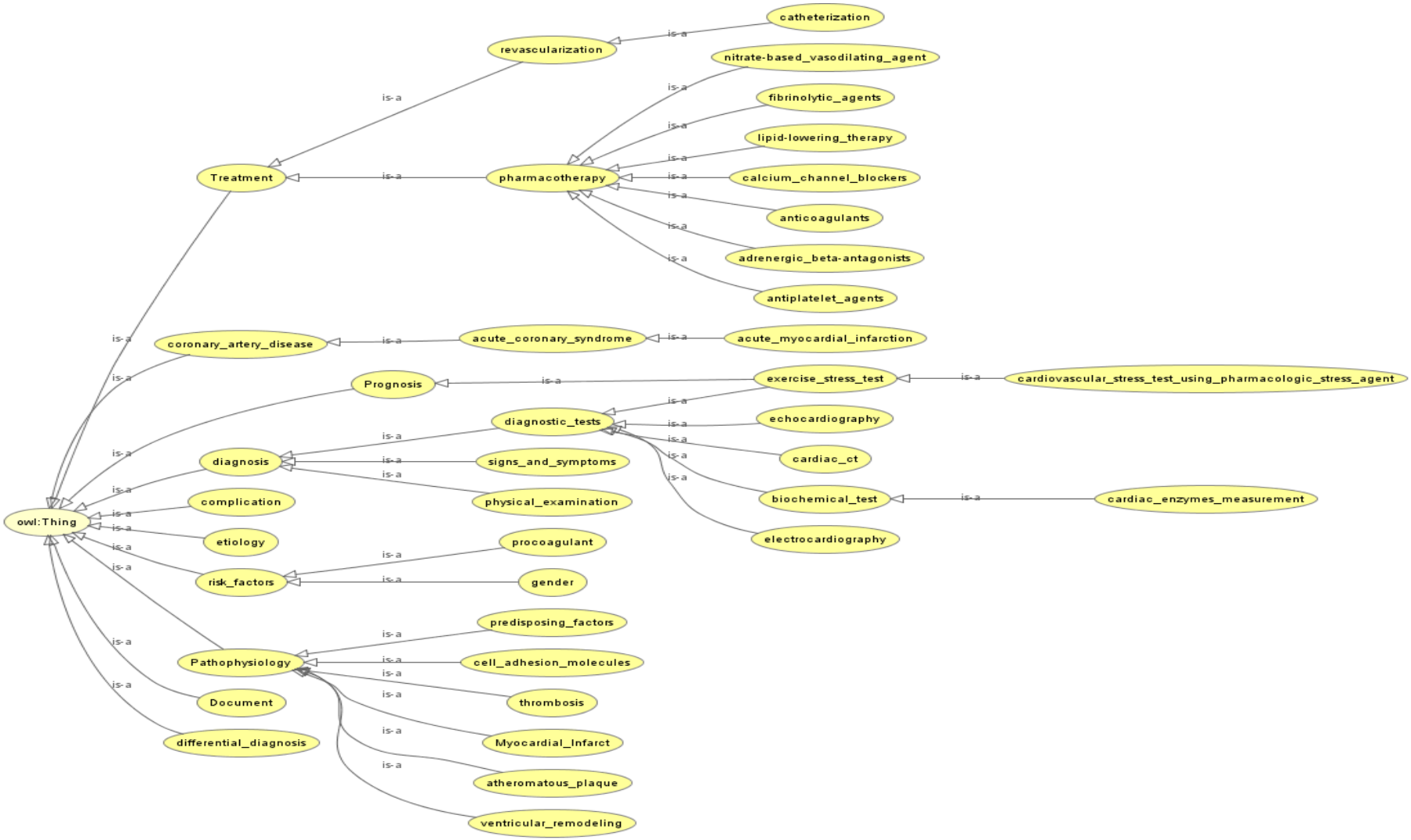
Αφού ο γιατρός μας έδωσε τα παραπάνω συστατικά που χρειαζόμασταν για τη χρήση της οντολογίας, αλλά και κατά τη διάρκεια αυτής της φάσης, αρχίσαμε να εισάγουμε τα δεδομένα μας μέσω Protégé (Classes, Instances, Object Properties, Annotation Properties) για να καταλήξουμε σταδιακά στην τελική έκδοση της οντολογίας μας. Εδώ πρέπει να σημειωθεί πως η οντολογία αν και η πλειοψηφία τον εννοιών που συμπεριλάβαμε είναι παρμένη από το UMLS, χρειάζεται επιπλέον εμπλουτισμό, κυρίως όσον αφορά τα συνώνυμα των κλάσεων και instances, πράγμα το οποίο είναι πολύ χρονοβόρα διαδικασία και επίσης χρειάζεται την συμβολή επιπλέον εξειδικευμένων γιατρών του πεδίου.

Στη συνέχεια αποδίδουμε την οντολογία που δημιουργήσαμε κατά τη διάρκεια της μεταπτυχιακής διατριβής.

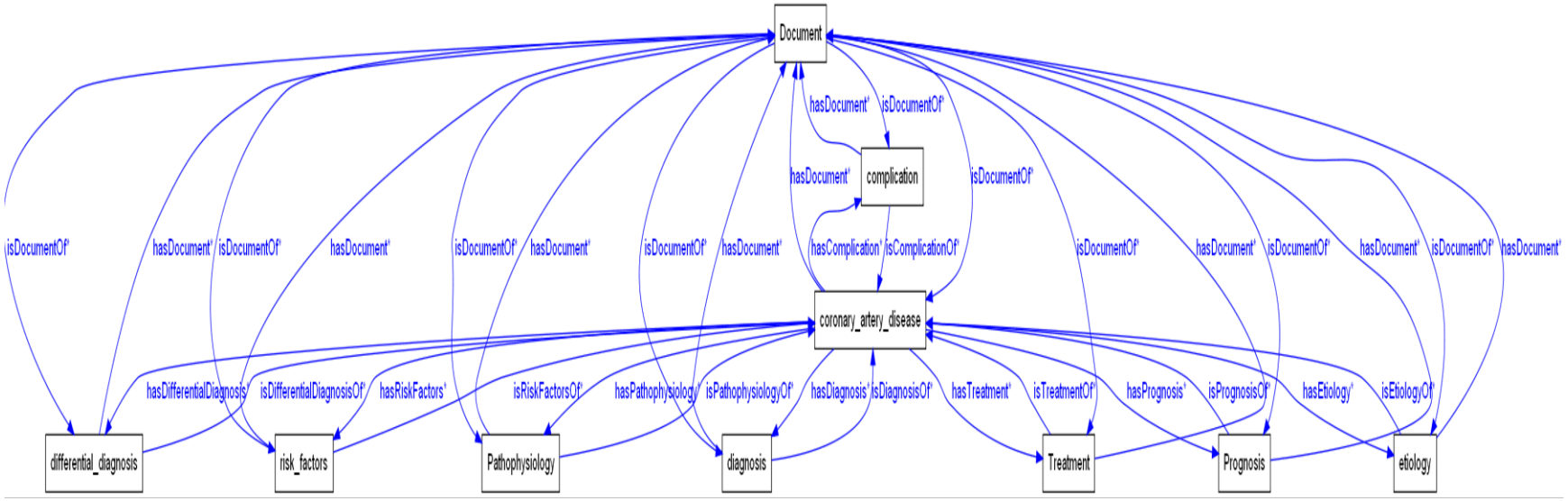
Στο σχήμα 6.8 βλέπουμε μια δεντρική αναπαράσταση της οντολογίας στο πεδίο των καρδιαγγειακών νοσημάτων, την ιεραρχία των κλάσεων και των υποκλάσεων τους.

Στο σχήμα 6.9 βλέπουμε πάλι μια δενδρική δομή των ριζικών κλάσεων της οντολογίας μας, μόνο που εδώ παρουσιάζουμε και τις σχέσεις (object properties) που έχουμε δημιουργήσει μεταξύ τους. Οι Σχέσεις αυτές μέσω της κληρονομικότητας που ισχύει στις οντολογίες, θα κληρονομηθούν και μεταξύ των υποκλάσεων των ριζικών κλάσεων.

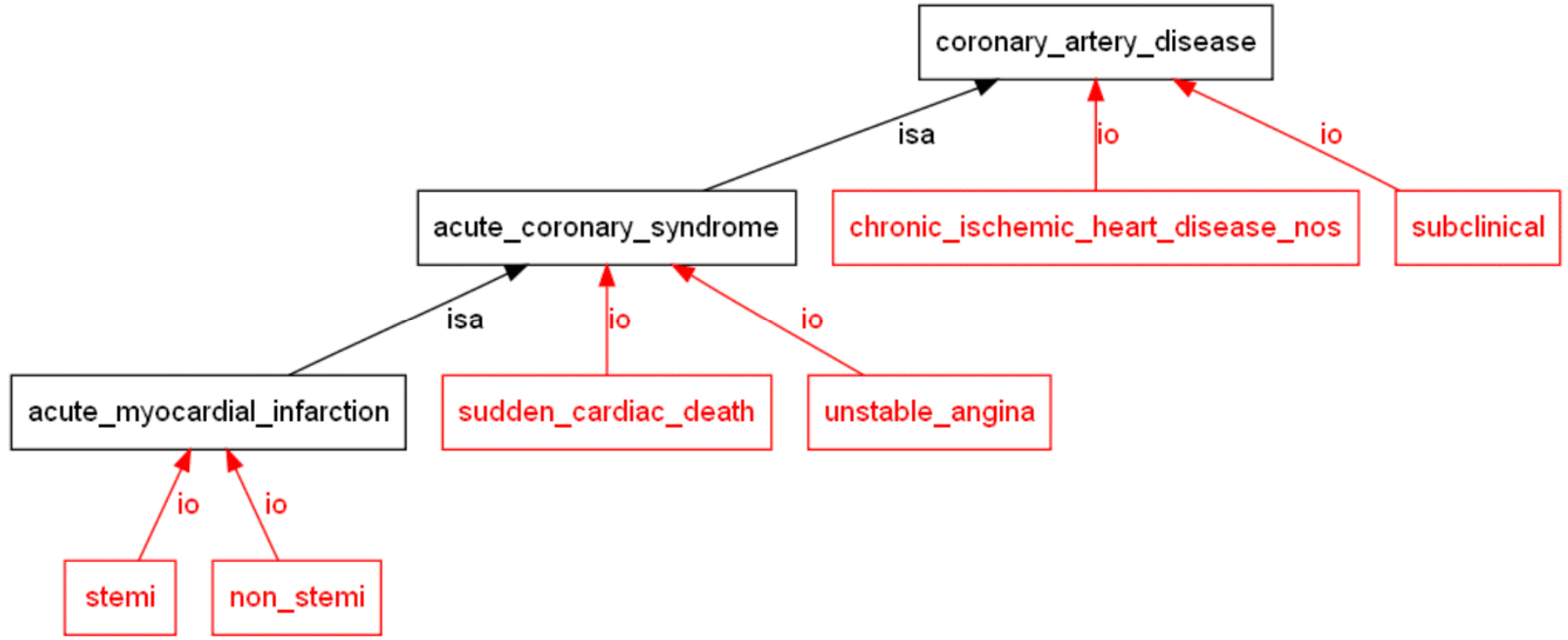
Τέλος στα Σχήματα 6.10 έως 6.46 δείχνουμε τα instances που εμπεριέχονται στην εκάστοτε κλάση/υποκλάση της οντολογίας μας.



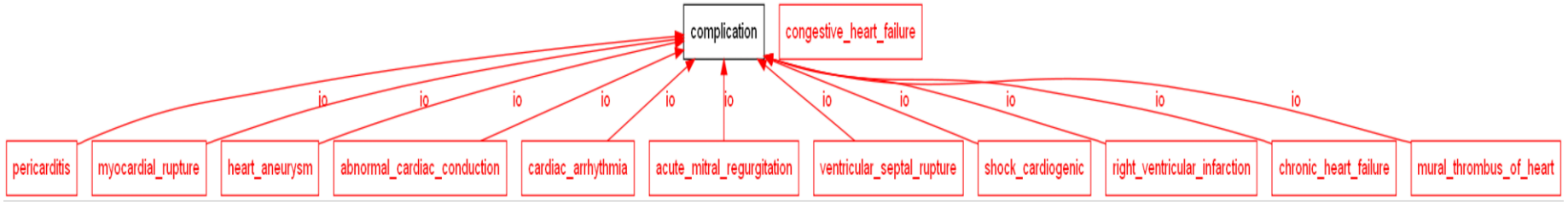
Σχήμα 6.8 Η ιεραρχική δομή της οντολογίας.



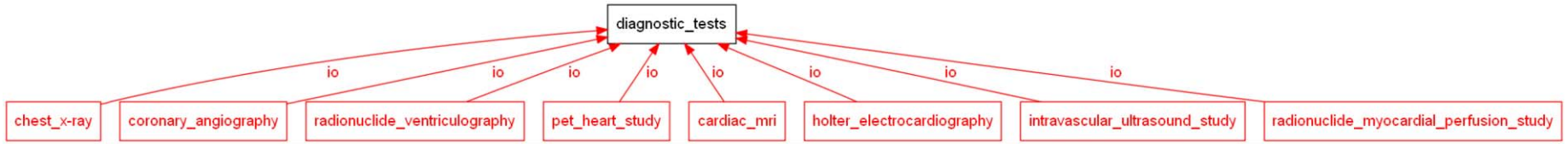
Σχήμα 6.9 Η Ιεραρχική δομή των ριζικών κλάσεων και οι σχέσεις μεταξύ τους.



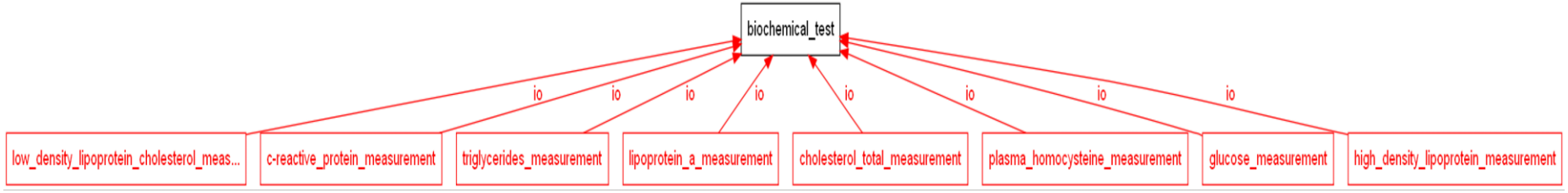
Σχήμα 6.10 της κλάσης Coronary artery disease και των υποκλάσεων της.



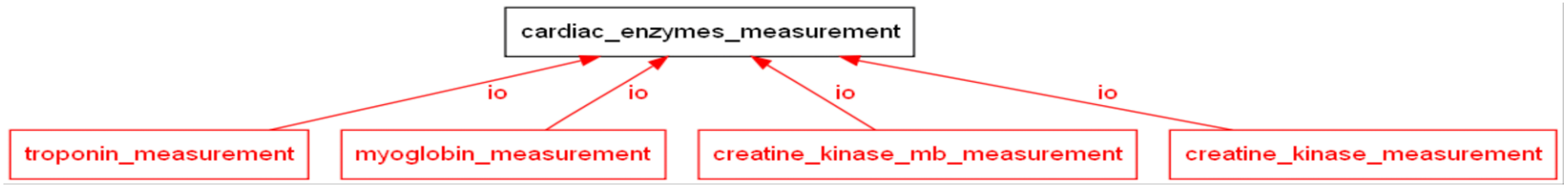
Σχήμα 6.11 Instances της κλάσης Complication.



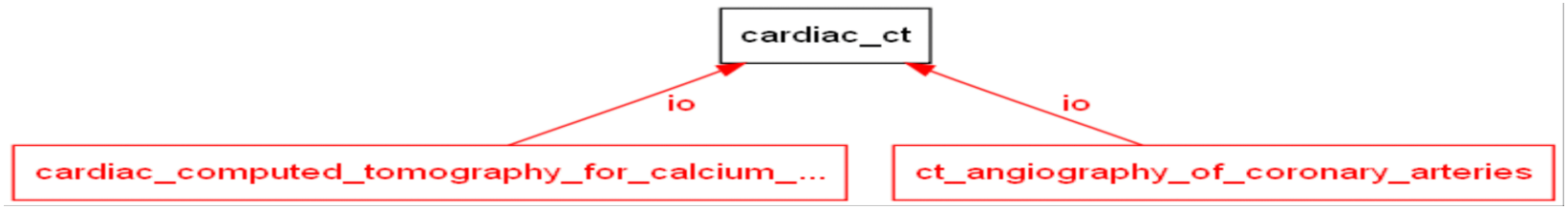
Σχήμα 6.12 Instances της υποκλάσης diagnostic tests.



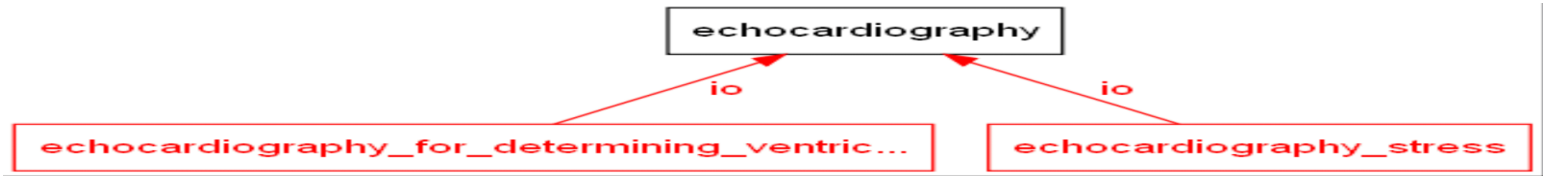
Σχήμα 6.13 Instances της υποκλάσης biochemical test.



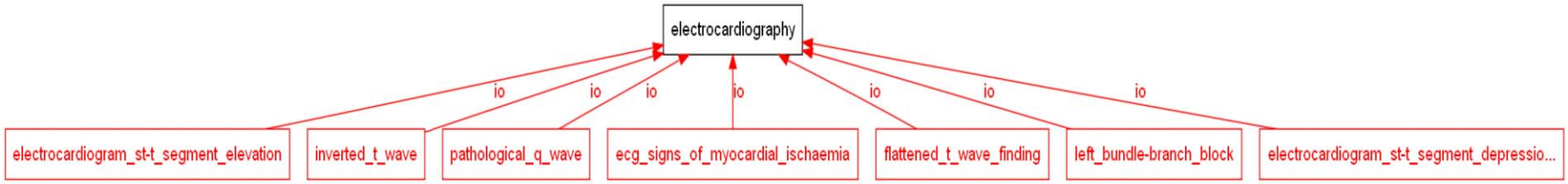
Σχήμα 6.14 Instances της υποκλάσης cardiac enzymes measurement.



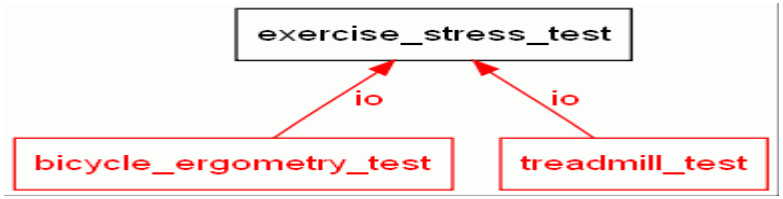
Σχήμα 6.15 Instances της υποκλάσης cardiac ct.



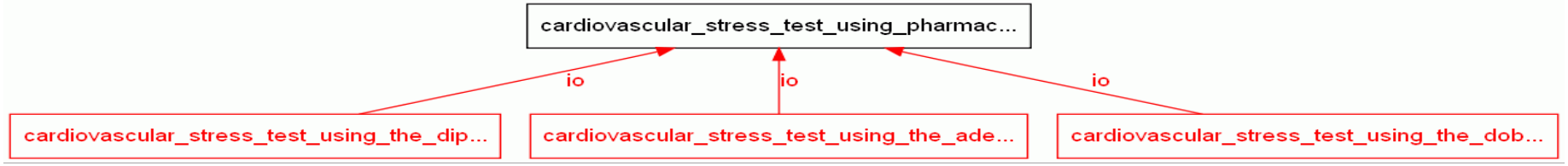
Σχήμα 6.16 Instances της υποκλάσης echocardiography.



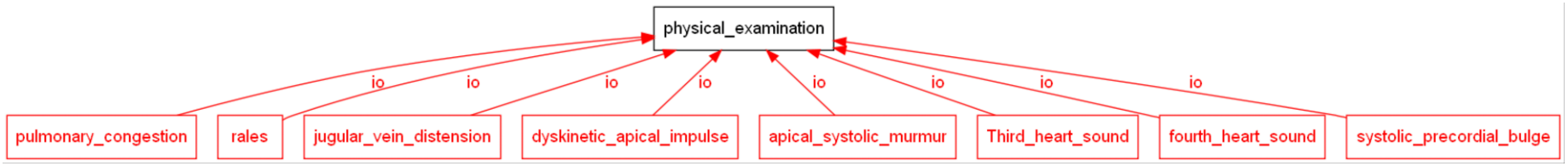
Σχήμα 6.17 Instances της υποκλάσης electrocardiography.



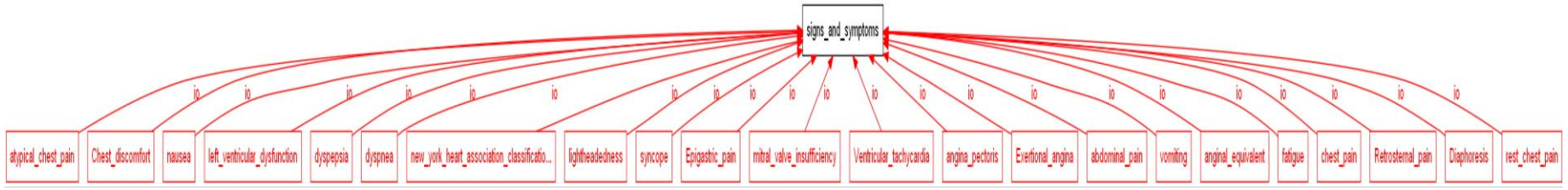
Σχήμα 6.18 της υποκλάσης exercise stress test.



Σχήμα 6.19 Instances της υποκλάσης cardiovascular stress test using pharmacologic stress agent.

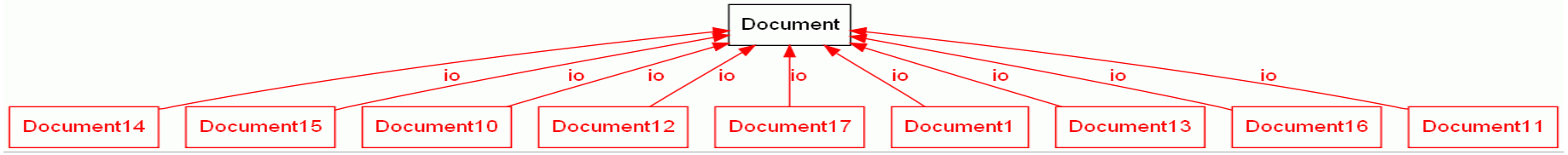


Σχήμα 6.20 Instances της υποκλάσης Physical examination.

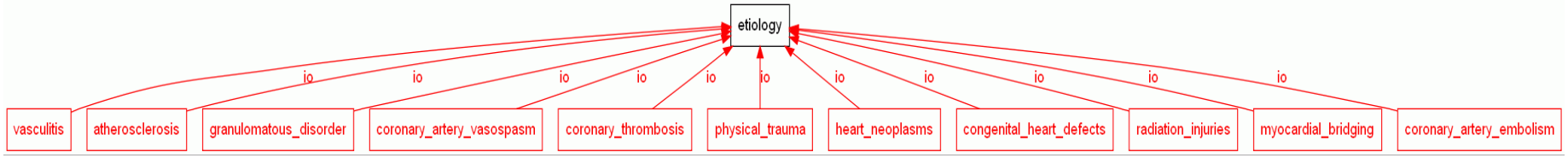


Σχήμα 6.21 Instances της υποκλάσης signs and symptoms.

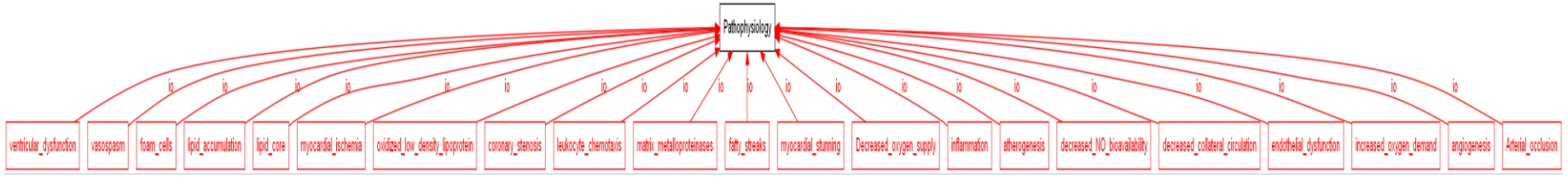
Σχήμα 6.22 Instances της κλάσης Differential diagnosis.



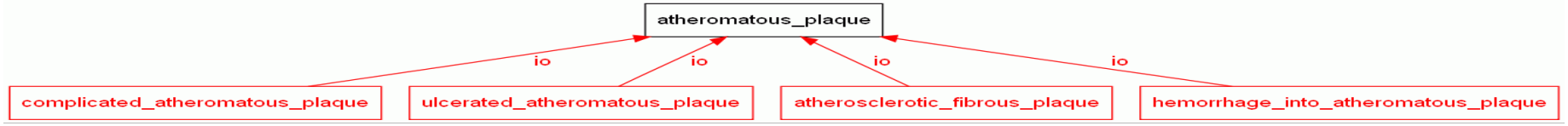
Σχήμα 6.23 Instances της κλάσης Document.



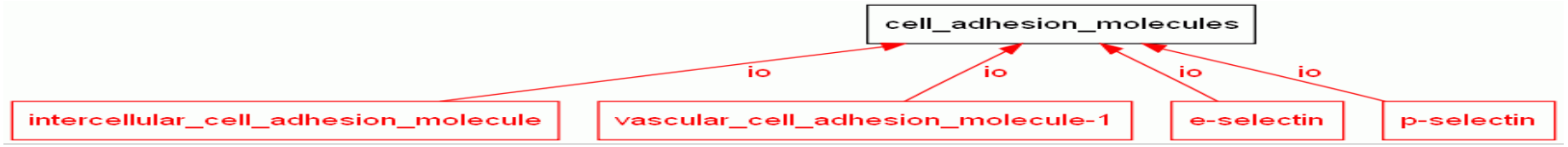
Σχήμα 6.24 Instances της κλάσης Etiology.



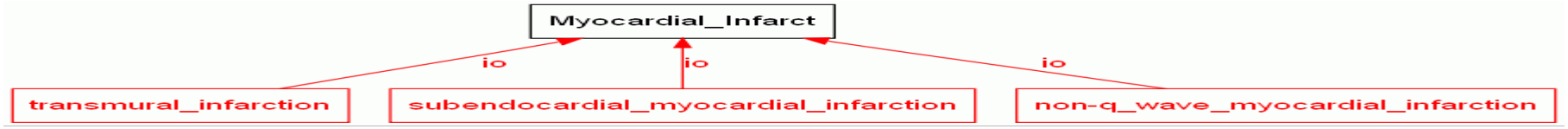
Σχήμα 6.25 Instances της κλάσης Pathophysiology.



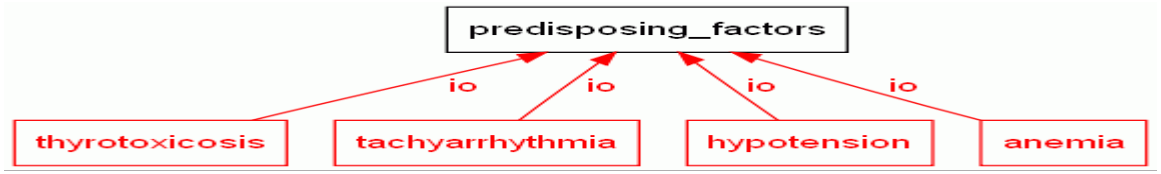
Σχήμα 6.26 Instances της υποκλάσης Atheromatous Plaque.



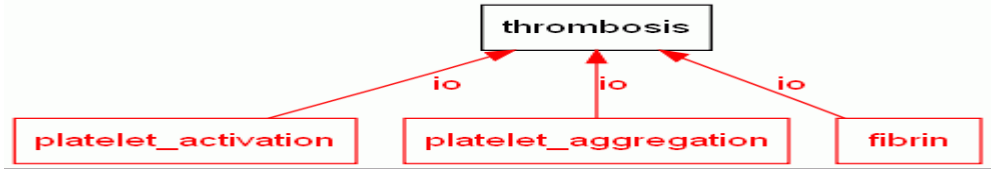
Σχήμα 6.27 Instances της υποκλάσης Cell adhesion molecules.



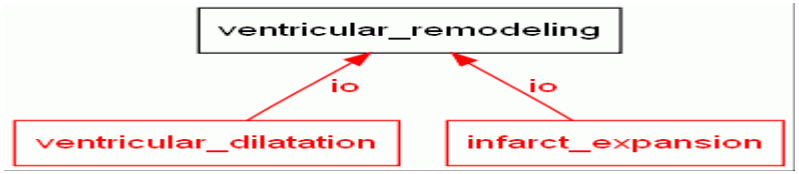
Σχήμα 6.28 Instances της υποκλάσης Myocardial Infarct.



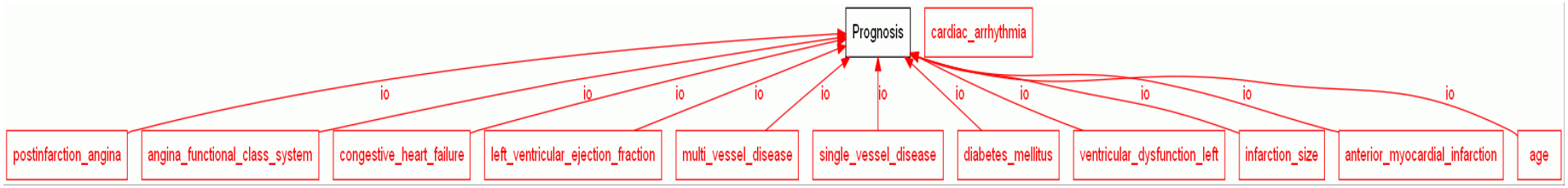
Σχήμα 6.29 Instances της υποκλάσης Predisposing factors.



Σχήμα 6.30 Instances της υποκλάσης Thrombosis.

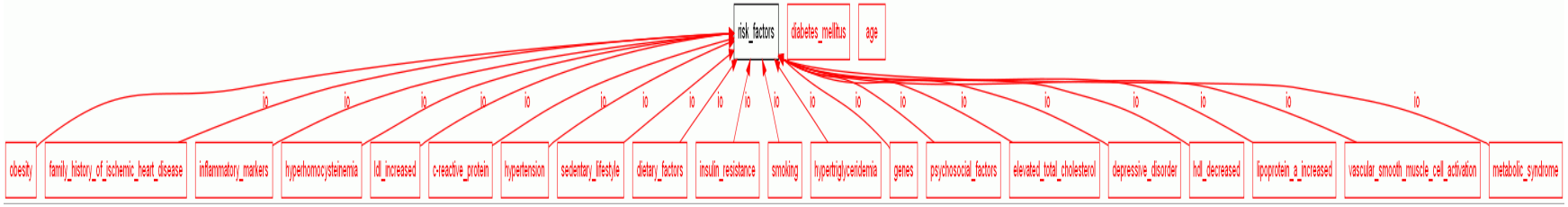


Σχήμα 6.31 Instances της υποκλάσης Ventricular remodeling.

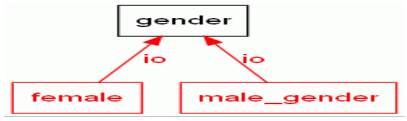


Σχήμα 6.32 Instances της κλάσης Prognosis.

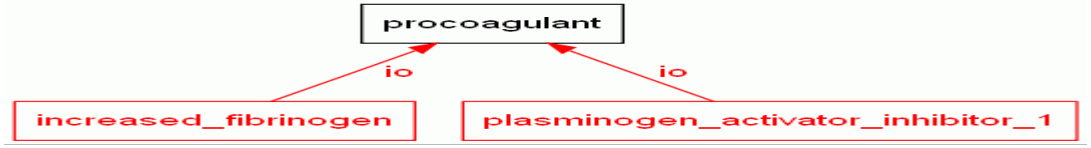
Το Prognosis έχει σαν υποκλάση το exercise stress test που τα instances του δόθηκαν παραπάνω λόγω του ότι είναι και υποκλάση του diagnostic tests.



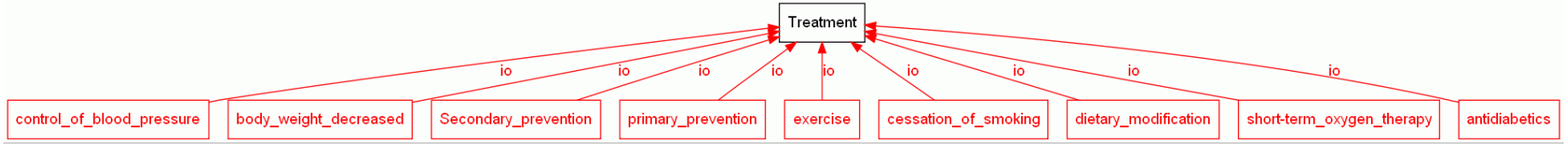
Σχήμα 6.33 Instances της κλάσης Risk factors.



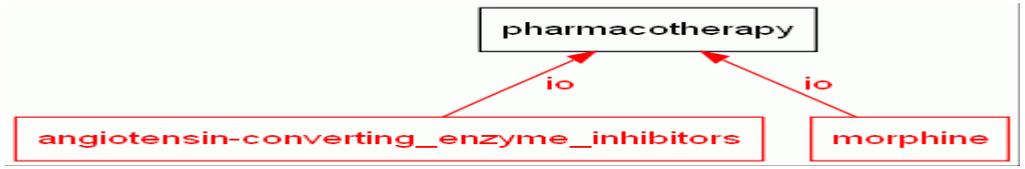
Σχήμα 6.34 Instances της υποκλάσης Gender.



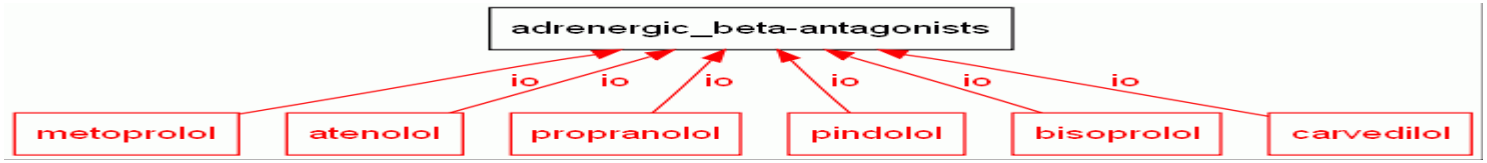
Σχήμα 6.35 Instances της υποκλάσης Procoagulant.



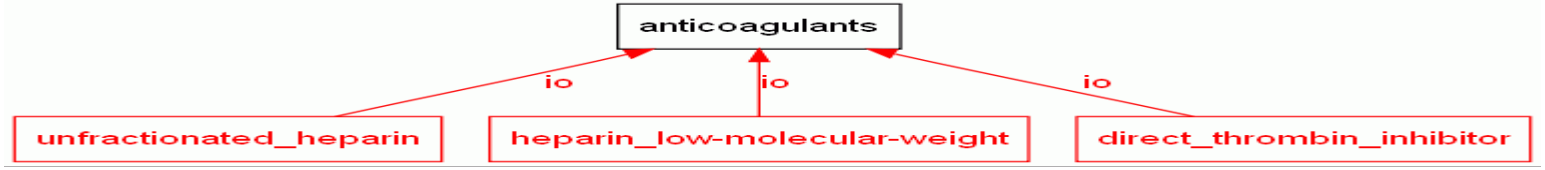
Σχήμα 6.36 Instances της κλάσης Treatment.



Σχήμα 6.37 Instances της υποκλάσης Pharmacotherapy.

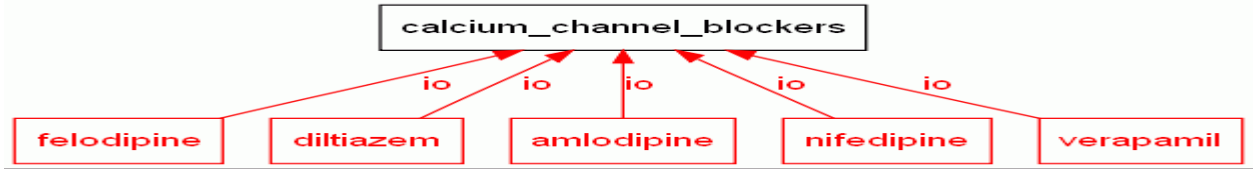


Σχήμα 6.38 Instances της υποκλάσης Adrenergic beta-antagonists.

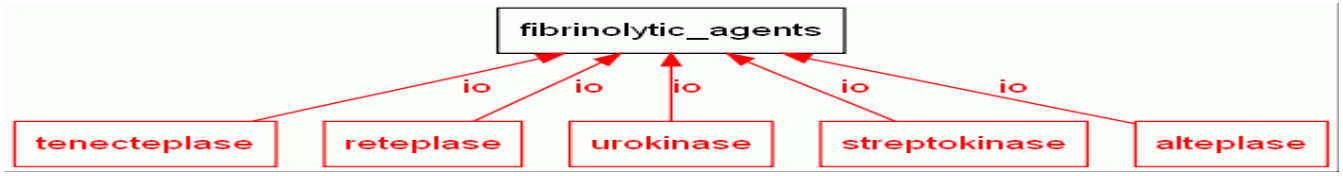


Σχήμα 6.39 Instances της υποκλάσης Anticoagulants.

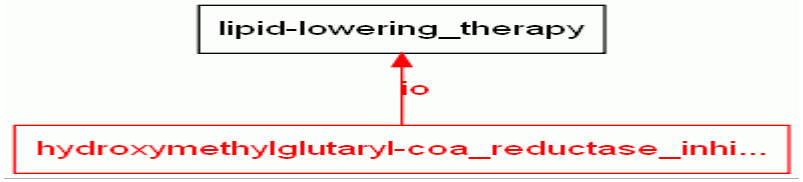
Σχήμα 6.40 Instances της υποκλάσης Antiplatelet agents.



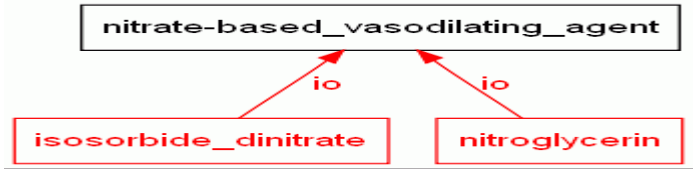
Σχήμα 6.41 Instances της υποκλάσης Calcium channel blockers.



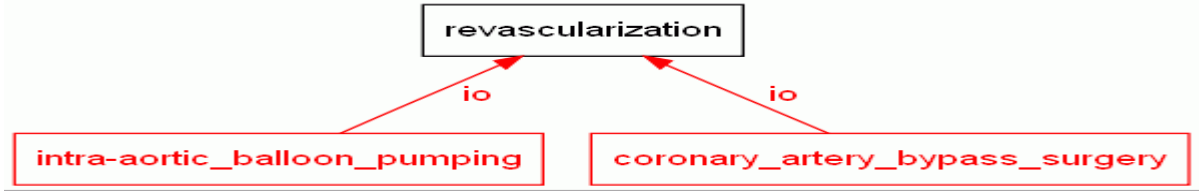
Σχήμα 6.42 Instances της υποκλάσης Fibrinolytic agents.



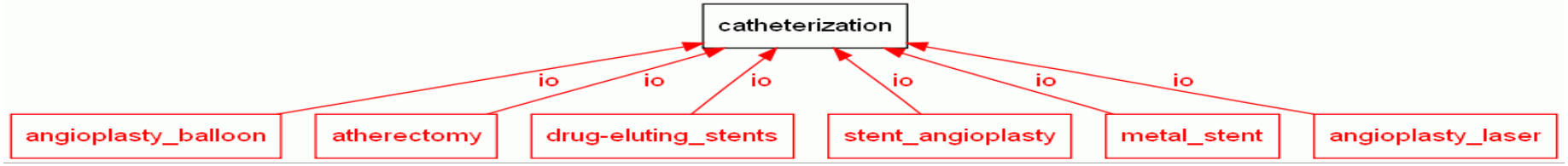
Σχήμα 6.43 Instances της υποκλάσης Lipid-lowering therapy.



Σχήμα 6.44 Instances της υποκλάσης Nitrate-based vasodilating agent.



Σχήμα 6.45 Instances της υποκλάσης Revascularization.



Σχήμα 6.46 Instances της υποκλάσης Catheterization.

ΚΕΦΑΛΑΙΟ 7. ΣΥΣΤΗΜΑ ΑΝΑΚΤΗΣΗΣ ΚΕΙΜΕΝΩΝ ΜΕ ΧΡΗΣΗ ΟΝΤΟΛΟΓΙΑΣ

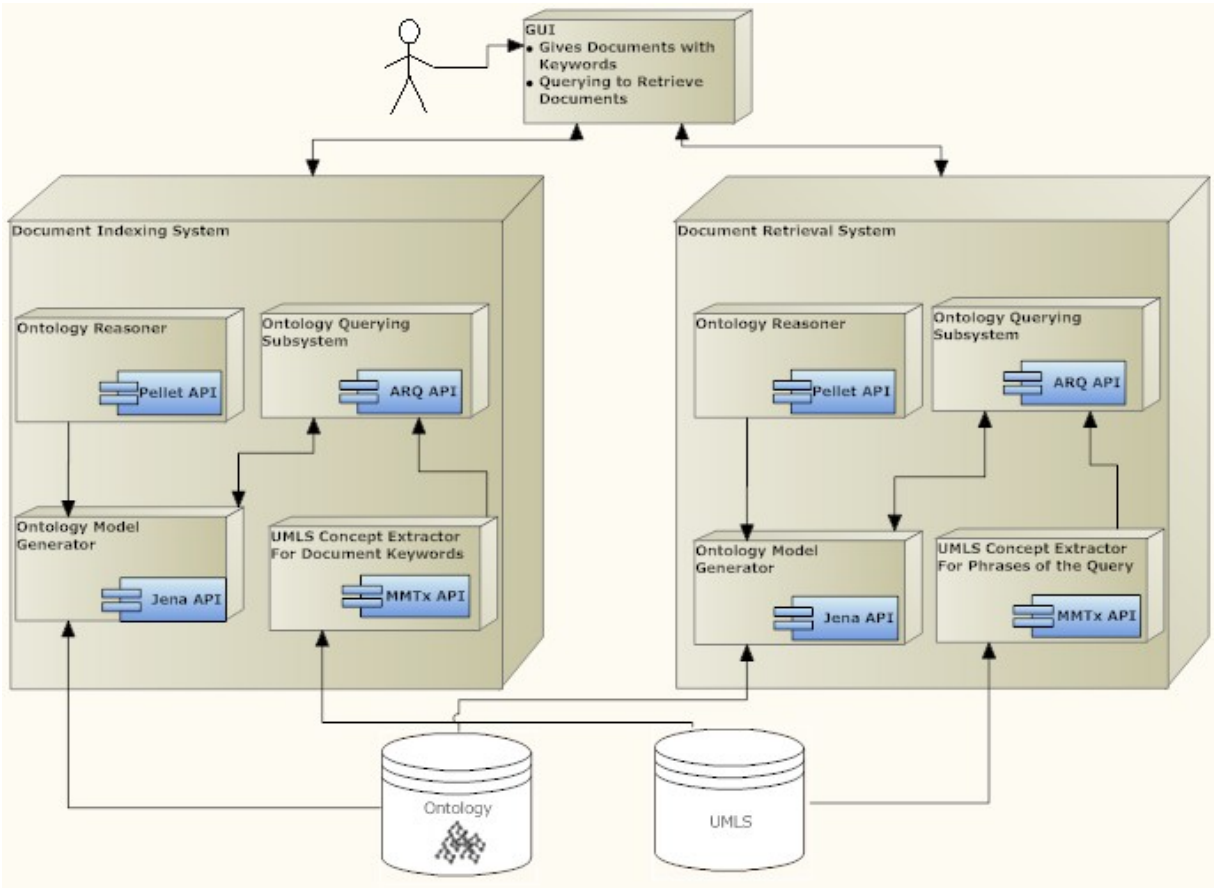
7.1. Περιγραφή του Συστήματος

Για τη χρήση της οντολογίας στο πεδίο των καρδιαγγειακών νοσημάτων, επιλέξαμε τη σχεδίαση και ανάπτυξη ενός συστήματος που ως σκοπό έχει την ανάκτηση, με σημασιολογικό τρόπο, εγγράφων του συγκεκριμένου πεδίου.

Κάποιος χρήστης μπορεί μέσω του συστήματος αυτού, να εισάγει ένα σύνολο εγγράφων επάνω στο οποίο μπορεί να πραγματοποιήσει αναζητήσεις σε κάποια μετέπειτα χρονική στιγμή. Το σύστημα αναλύει τα έγγραφα σύμφωνα με τις λέξεις κλειδιά που αυτά συμπεριλαμβάνουν, και εντοπίζει τις καταλληλότερες έννοιες της οντολογίας μας, με τις οποίες το εκάστοτε έγγραφο πρέπει να συνδεθεί. Ο χρήστης είναι σε θέση να κάνει αναζήτηση εγγράφων εκφράζοντας σε φυσική γλώσσα το ερώτημά του, και παίρνει ως έξοδο μια σειρά από έγγραφα που σχετίζονται με την ερώτησή του. Το σύστημα εκμεταλλεύεται από τη μια πλευρά την οντολογία, που αποτελεί την ραχοκοκαλιά του συστήματος, τα συστατικά της (classes, instances) έχουν παρθεί από το UMLS [72] και η οποία με κάποιο τρόπο έχει αποθηκεύσει στην εσωτερική της δομή τα έγγραφα επάνω στα οποία μπορεί να γίνει η αναζήτηση. Από την άλλη εκμεταλλεύεται έναν φραστικό αναλυτή (MMTx [73]), μέσω του οποίου μπορεί να παίρνει τις έννοιες του UMLS που συσχετίζονται είτε με τις λέξεις κλειδιά των εγγράφων, είτε με την ερώτηση του χρήστη, ώστε να δεικτοδοτήσει με τον καλύτερο τρόπο τα έγγραφα με έννοιες της οντολογίας μας, και να εκτελέσει επίσης την αναζήτηση με βάση τις έννοιες αυτές.

7.2. Αρχιτεκτονική του Συστήματος και Τρόπος Λειτουργίας του

Το σύστημα υλοποιεί δύο διαδικασίες. Η μια είναι η δεικτοδότηση των εγγράφων, μέσω σχέσεων με κόμβους της οντολογίας, ανάλογα με την αντιστοίχηση που έχει γίνει μεταξύ λέξεων κλειδιών και συστατικών της οντολογίας. Η άλλη είναι η ανάκτηση εγγράφων μέσω ερωτήσεων φυσικής γλώσσας του χρήστη. Τα δομικά συστατικά του συστήματος φαίνονται στο Σχήμα 7.1.



Σχήμα 7.1 Τα Δομικά Συστατικά του Συστήματος Ανάκτησης Εγγράφων.

Όπως φαίνεται στο σχήμα 7.1. έχουμε τα εξής συστατικά του συστήματος:

- Σύστημα Δεικτοδότησης Εγγράφων:
    - Μια γραφική διαπροσωπία χρήστη (GUI), μέσω της οποίας ο χρήστης έρχεται σε επαφή με το σύστημα και το τροφοδοτεί με μια λίστα εγγράφων και τις σχετικές με αυτά λέξεις κλειδιά.

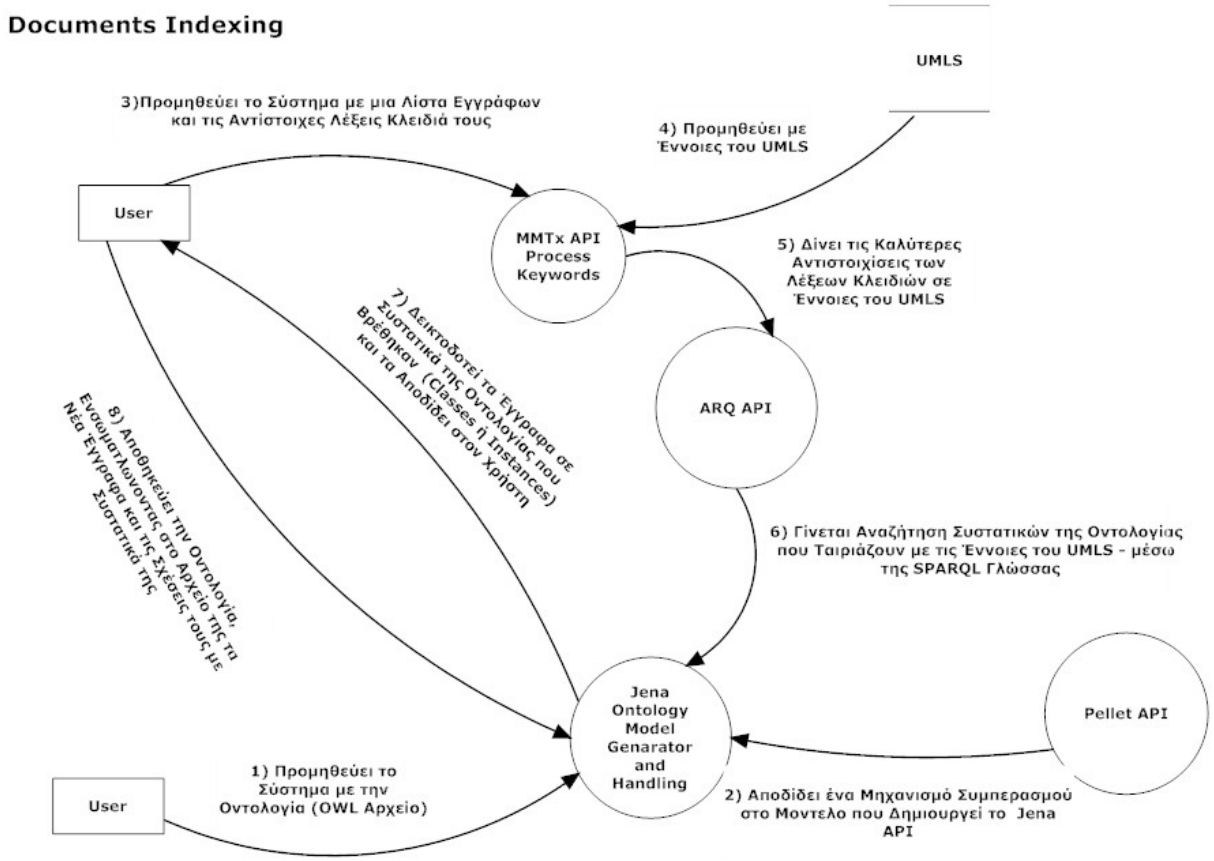
- Την Οντολογία. Είναι εκφρασμένη στη γλώσσα οντολογιών OWL και αποθηκευμένη σε ένα OWL αρχείο.
- Το UMLS. Είναι εγκατεστημένο τοπικά και τροφοδοτεί το σύστημα με τις έννοιες που περιλαμβάνει.
- Το MMTx API. Διαπροσωπία που χρησιμοποιείται στη διαδικασία αντιστοίχησης λέξεων κλειδιών και εννοιών του UMLS.
- Το Jena API. Διαπροσωπία που μοντελοποιεί την οντολογία στη μνήμη του υπολογιστή και επιτρέπει στον προγραμματιστή να τη διαχειριστεί (εντοπισμός συστατικών της και δημιουργία νέων).
- Το Pellet API. Διαπροσωπία για την ενσωμάτωση σε συστήματα που κάνουν χρήση οντολογιών, ενός μηχανισμού συμπερασμού, ώστε να είναι δυνατόν να εκμαιεύεται υπονοούμενη πληροφορία.
- Το ARQ API. Εφοδιάζει τον προγραμματιστή με μια διαπροσωπία για τη χρήση της γλώσσας επερωτήσεων οντολογιών, SPARQL.

- Σύστημα Ανάκτησης Εγγράφων:
  - Μια γραφική διαπροσωπία χρήστη (GUI), μέσω της οποίας ο χρήστης έρχεται σε επαφή με το σύστημα και μπορεί να εκτελέσει ερωτήματα σε φυσική γλώσσα, με σκοπό την ανάκτηση εγγράφων σχετικών με την ερώτησή του.
  - Την Οντολογία. Είναι εκφρασμένη στη γλώσσα οντολογιών OWL και αποθηκευμένη σε ένα OWL αρχείο.
  - Το UMLS. Είναι εγκατεστημένο τοπικά και τροφοδοτεί το σύστημα με τις έννοιες που περιλαμβάνει.
  - Το MMTx API. Διαπροσωπία που χρησιμοποιείται στη διαδικασία ανάλυσης της ερώτησης του χρήστη. Εντοπίζει τις συστατικές φράσεις του ερωτήματος και αποδίδει την καλύτερη αντιστοίχηση εννοιών του UMLS στην εκάστοτε φράση.
  - Το Jena API. Διαπροσωπία που μοντελοποιεί την οντολογία στη μνήμη του υπολογιστή και επιτρέπει στον προγραμματιστή να τη διαχειριστεί (εντοπισμός συστατικών της και δημιουργία νέων).
  - Το Pellet API. Διαπροσωπία για την ενσωμάτωση σε συστήματα που κάνουν χρήση οντολογιών, ενός μηχανισμού συμπερασμού, ώστε να είναι δυνατόν να εκμαιεύεται υπονοούμενη πληροφορία.

- Το ARQ. Εφοδιάζει τον προγραμματιστή με μια διαπροσωπία για τη χρήση της γλώσσας επερωτήσεων οντολογιών, SPARQL.

7.2.1. Αναλυτική Περιγραφή Λειτουργίας του Συστήματος

Η ροή των δεδομένων στο σύστημα δεικτοδότησης εγγράφων σε συστατικά της οντολογίας (Document Indexing), παρουσιάζεται στο Σχήμα 7.2.



Σχήμα 7.2 Ροή Δεδομένων κατά τη Λειτουργία Δεικτοδότησης Εγγράφων από την Οντολογία.

1) Το σύστημα παίρνει ως είσοδο μια οντολογία, εκφρασμένη σε OWL (OWL file) η οποία είναι και η ραχοκοκαλιά του συστήματός μας. Μέσω του Jena API μοντελοποιείται η οντολογία στη μνήμη του υπολογιστή, και έτσι μπορούμε να τη διαχειριστούμε. Με τον όρο διαχείριση της οντολογίας, εννοούμε τον εντοπισμό συστατικών της οντολογίας (classes, instances, properties κ.α.) και τρόπους χειρισμού

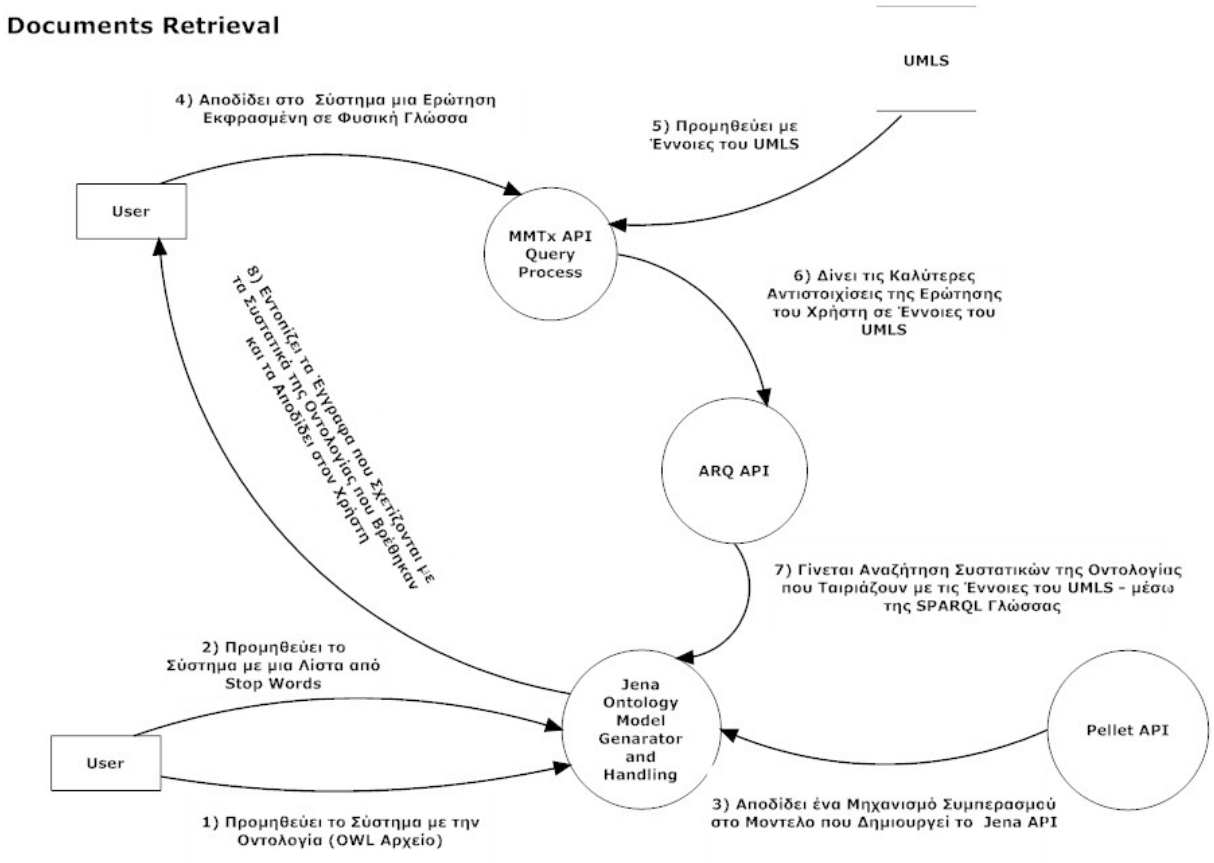
αυτών μέσω κώδικα java. Επίσης το Jena API επιτρέπει την εισαγωγή νέων συστατικών οντολογίας (π.χ. τα κείμενα προς δεικτοδότηση, και σχέσεις μεταξύ αυτών και συστατικών της οντολογίας που υπάρχουν ήδη σε αυτή).

2) Γίνεται χρήση του Pellet reasoner για να μπορεί να υπάρξει μηχανισμός συμπερασμού στο μοντέλο της οντολογίας που θα δημιουργηθεί και να εξαχθεί υπονοούμενη πληροφορία από την ήδη υπάρχουσα.
3) Ως είσοδος δίνεται επίσης ένα αρχείο κειμένου (TXT file). Η κάθε γραμμή του αρχείου κειμένου, αντιστοιχεί σε ένα έγγραφο, έχοντας στην πρώτη θέση ένα μοναδικό προσδιοριστή, ακολουθεί ο τίτλος του κειμένου και μια λίστα από λέξεις κλειδιά που χαρακτηρίζουν το κείμενο. Το κάθε συστατικό της κάθε γραμμής διαχωρίζεται από τα υπόλοιπα με ένα κόμμα ",". Ένα παράδειγμα μιας γραμμής αυτού του αρχείου είναι το εξής "Document59,The potential clinical utility of intravascular ultrasound guidance in patients undergoing percutaneous coronary intervention with drug-eluting stents,Intravascular ultrasound,Drug-eluting stent,Percutaneous coronary intervention,". Ως έξοδο αποδίδει στην οθόνη του χρήστη τις λέξεις κλειδιά του κάθε κειμένου, με τις αντιστοιχίσεις τους σε έννοιες της οντολογίας μας, εάν αυτές υπάρχου. Εάν δεν υπάρχουν, ειδοποιεί το χρήστη πως για μια συγκεκριμένη λέξη κλειδί απέτυχε η αντιστοίχηση στην οντολογία, και παροτρύνει
4) Το UMLS τροφοδοτεί το MMTx με έννοιες του UMLS, ούτως ώστε αυτό με τη σειρά του να μπορέσει να κάνει μια αντιστοίχηση μεταξύ λέξεων κλειδιών και εννοιών του UMLS.
5) Το MMTx αναλύει την κάθε λέξη κλειδί που έχει δοθεί για το κάθε έγγραφο και αποδίδει την καλύτερη αντιστοίχηση μεταξύ λέξεων κλειδιών και εννοιών του UMLS που τις εκφράζουν.
6) Αφού έχει γίνει η αντιστοίχηση μεταξύ λέξεων κλειδιών και εννοιών του UMLS, γίνεται χρήση του ARQ API μέσω του οποίου μπορούμε να κάνουμε επερωτήσεις στο μοντέλο της οντολογίας με τη γλώσσα επερωτήσεων οντολογιών SPARQL. Ουσιαστικά στο βήμα αυτό προσπαθούμε να βρούμε αντιστοίχηση μεταξύ των λέξεων κλειδιών των εγγράφων και συστατικών της οντολογίας. Γίνεται αναζήτηση στην οντολογία μας, για έννοιες που είτε έχουν ως ετικέτα τους, είτε ως συνώνυμό τους τις έννοιες του UMLS που επιστράφηκαν από το MMTx.. Αυτό επιτυγχάνεται αφού η οντολογία κάνει χρήση εννοιών του UMLS, οι λέξεις κλειδιά μετά την επεξεργασία

τους από το MMTx είναι εκφρασμένες και αυτές μέσω εννοιών του UMLS, οπότε αναζητούμε στην οντολογία για κοινά πράγματα.

7) Πλέον τα έγγραφα δεικτοδοτούνται στην οντολογία, ως instances της κλάσης Document. Επίσης έχει γίνει σύνδεση των εγγράφων με συστατικά της οντολογίας, τα οποία εκφράζουν τις λέξεις κλειδιά τους. Η σύνδεση των εγγράφων με συστατικά της οντολογίας εκφράζεται μέσω των σχέσεων οντολογίας "hasDocument" και "isDocumentOf" που ορίζονται μέσω των Document instances και των αντίστοιχων instances της οντολογίας που εκφράζουν τις λέξεις κλειδιά των εγγράφων. Η όλη αυτή διαδικασία γίνεται μέσω του Jena API. Αφού έχουν ολοκληρωθεί τα παραπάνω, αποδίδονται στην οθόνη του χρήστη η αντιστοιχίσεις των λέξεων κλειδιών του κάθε εγγράφου με έννοιες που εντοπίστηκαν στην οντολογία. Σε περίπτωση που δεν εντοπιστεί μια τέτοια αντιστοίχηση, τότε το σύστημα βγάζει ως έξοδο, ενημέρωση του χρήστη για αποτυχία αντιστοίχησης μιας λέξεως κλειδί με κάποιο συστατικό της οντολογίας, καθώς και παροτρύνει για εμπλουτισμό της.
8) Επίσης μετά την όλη διαδικασία ο χρήστης πρέπει να κάνει αποθήκευση της οντολογίας, ούτως ώστε να ενημερωθεί το αρχείο της (OWL file) με τα έγγραφα που εισήχθησαν και τις κατάλληλές συσχετίσεις τους με έννοιες της οντολογίας.

Η ροή των δεδομένων στο σύστημα ανάκτησης εγγράφων (Document Retrieval), παρουσιάζεται στο Σχήμα 7.3.



Σχήμα 7.3 Δεδομένων κατά τη Διαδικασία Ανάκτησης Εγγράφων.

1) Το σύστημα παίρνει ως είσοδο μια οντολογία, εκφρασμένη σε OWL (OWL file). Μέσω του Jena API μοντελοποιείται η οντολογία στη μνήμη του υπολογιστή, και έτσι μπορούμε να τη διαχειριστούμε.
2) Το MMTx κατά τη διαδικασία ανάλυσης του ερωτήματος του χρήστη, αναγνωρίζει, σαν ξεχωριστές φράσεις, κάποιες λέξεις ή φράσεις του ερωτήματος του χρήστη, που δεν έχουν άμεση σχέση με το πεδίο και που χρησιμοποιούνται στην αγγλική γλώσσα, τις επονομαζόμενες "stop words",. Παράδειγμα τέτοιων λέξεων είναι οι "this", "was", "what", "where", "with", "the" κ.α. Λόγω του ότι αυτές οι φράσεις δεν πρέπει να ληφθούν υπόψη στη σημασιολογική αναζήτηση που κάνει το σύστημα, το εφοδιάζουμε με μια λίστα τέτοιων λέξεων [84] ώστε να τις παραβλέπει.
3) Γίνεται χρήση του Pellet reasoner για να μπορεί να υπάρξει μηχανισμός συμπερασμού στο μοντέλο της οντολογίας που θα δημιουργηθεί και να εξαχθεί υπονοούμενη πληροφορία από την ήδη υπάρχουσα.

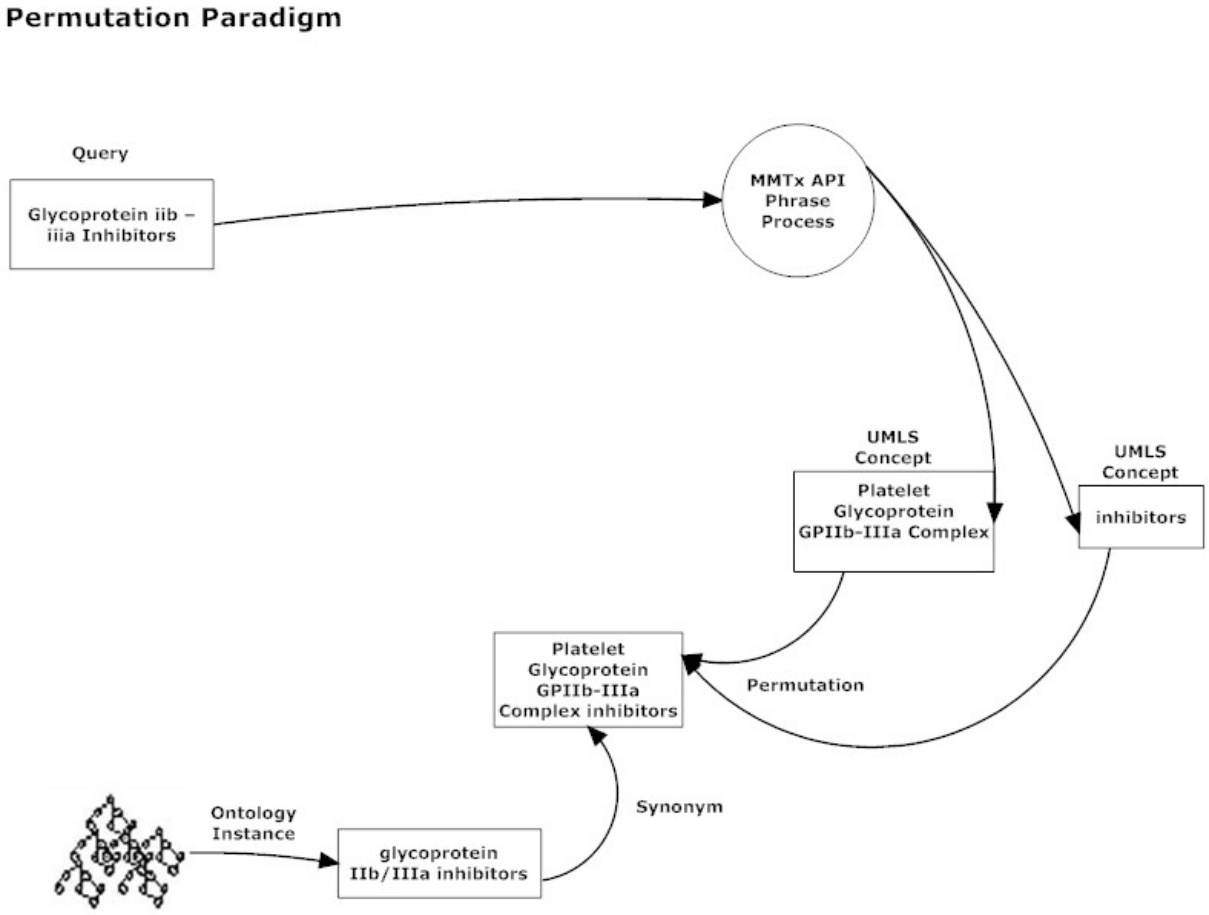
4) Ο χρήστης είναι πλέον σε θέση να αποδώσει κάποιο ερώτημα στο σύστημα, μέσω του οποίου αναζητά κάποια έγγραφα που συσχετίζονται με αυτό και που σε προηγούμενη φάση είχαν εισαχθεί στο σύστημα. Το ερώτημα μπορεί να είναι εκφρασμένο σε φυσική γλώσσα.
5) Το UMLS τροφοδοτεί το MMTx με έννοιες του UMLS, ούτως ώστε αυτό με τη σειρά του να μπορέσει να κάνει μια αντιστοίχηση μεταξύ των φράσεων που αποτελούν το ερώτημα του χρήστη και εννοιών του UMLS.
6) Ο φραστικός αναλυτής (MMTx) επιστρέφει για κάθε ερώτηση του χρήστη, τις έννοιες του UMLS οι οποίες εκφράζουν με όσο το δυνατόν καλύτερο τρόπο την εκάστοτε ερώτηση. Ουσιαστικά, σπάει την ερώτηση του χρήστη σε φράσεις, και για κάθε φράση επιστρέφει τις κοντινότερες έννοιες του UMLS που την εκφράζουν. Στη συνέχεια, γίνεται αναζήτηση στην οντολογία μας, για έννοιες που είτε έχουν ως ετικέτα τους, είτε ως συνώνυμό τους τις έννοιες του UMLS που επιστράφηκαν από το MMTx. Έτσι τελικά, έχουμε στη διάθεσή μας ένα σύνολο εννοιών της οντολογίας μας, οι οποίες εκφράζουν το ερώτημα του χρήστη.
7) Αφού έχει εκφραστεί το ερώτημα του χρήστη με έννοιες του UMLS, γίνεται χρήση του ARQ API μέσω του οποίου μπορούμε να κάνουμε επερωτήσεις στο μοντέλο της οντολογίας με τη γλώσσα επερωτήσεων οντολογιών SPARQL. Ουσιαστικά στο βήμα αυτό προσπαθούμε να βρούμε αντιστοίχηση μεταξύ της ερώτησης του χρήστη και συστατικών της οντολογίας. Γίνεται αναζήτηση στην οντολογία μας, για έννοιες που είτε έχουν ως ετικέτα τους, είτε ως συνώνυμό τους, τις έννοιες του UMLS που επιστράφηκαν από το MMTx..
8) Πλέον έχουμε στη διάθεσή μας τα συστατικά της οντολογίας που εκφράζουν το ερώτημα του χρήστη. Τα συστατικά αυτά είναι συνδεδεμένα με κάποια έγγραφα (μέσω της σχέσης “hasDocument”) που εισάχθηκαν κατά τη διαδικασία δεικτοδότησης των εγγράφων στην οντολογία. Έτσι το σύστημα είναι σε θέση να εντοπίσει τα κοινά έγγραφα που συνδέονται με τα παραπάνω συστατικά της οντολογία, που ουσιαστικά αποτελούν την απάντηση στο ερώτημα του χρήστη και να τα αποδώσει ως έξοδό του.

Σε κάποιες περιπτώσεις στην οντολογίας, μπορεί να υπάρχουν έννοιες, που ο ειδικός του πεδίου (ιατρός) θεώρησε πως πρέπει να εισαχθούν στην περιγραφή του πεδίου, άσχετα με το αν αυτές οι έννοιες δεν υπήρχαν επακριβώς στο UMLS. Στην περίπτωση που μια

τέτοια έννοια, συσχετίζεται με λέξη κλειδί ενός εγγράφου, ή με μια φράση του ερωτήματος του χρήστη, το MMTx δεν θα μπορέσει να μας την επιστρέψει επακριβώς, ώστε να μπορέσει να γίνει επιτυχής αντιστοίχηση. Το πρόβλημα αυτό το ξεπερνάμε μερικώς, με δύο τρόπους.

- Ο πρώτος τρόπος, ελέγχει αν η ίδια η λέξη κλειδί ενός εγγράφου ή μια φράση του ερωτήματος, υπάρχει αυτούσια στην οντολογία μας, χωρίς να περάσει από επεξεργασία από το MMTx. Για παράδειγμα αν μια λέξη κλειδί για ένα έγγραφο ή μια φράση σε ένα ερώτημα είναι η "transmural infarction", το MMTx θα επιστρέψει δύο έννοιες ως την καλύτερη αντιστοίχηση στο UMLS, τις "transmural infarct" και "infarction". Στην οντολογία υπάρχει η έννοια "transmural infarction" και έχει την "transmural infarct" ως συνώνυμό της. Με το παράδειγμα αυτό αν δεν ελέγξουμε την οντολογία πριν την επεξεργασία της φράσης από το MMTx δεν θα μπορέσουμε να βρούμε κάποια αντιστοίχηση μεταξύ της φράσης και κάποιου συστατικού της οντολογίας μας, ακόμη και στην περίπτωση των μεταθέσεων που αναλύεται αμέσως μετά. Έτσι κάνοντας έλεγχο πριν την επεξεργασία μιας φράσης από το MMTx μπορούμε να εντοπίσουμε τέτοιες αντιστοιχίσεις.
- Ο δεύτερος τρόπος, στηρίζεται στο γεγονός πως το MMTx, για κάποιες λέξεις κλειδιά ή φράσεις, μπορεί να μην επιστρέψει μια μοναδική έννοια του UMLS που τα εκφράζει, αλλά ένα σύνολο από έννοιες, που σαν σύνολο αποτελούν (κατά την κρίση του MMTx) την καλύτερη αντιστοίχηση της λέξης κλειδί ή φράσης με έννοιες του UMLS. Στην περίπτωση αυτή παίρνουμε όλες τις δυνατές μεταθέσεις των εννοιών που επέστρεψε το UMLS χρησιμοποιώντας τον κώδικα από το [85] και κάνουμε αναζήτηση στην οντολογία, του όρου που προκύπτει από την κάθε μετάθεση. Παίρνουμε όλες τις δυνατές μεταθέσεις, γιατί ο ειδικός του πεδίου (ιατρός), μας έδωσε και εισάγαμε στην οντολογία τον όρο, με μία συγκεκριμένη σειρά συνδυασμού κάποιων λέξεων, ενώ δεν γνωρίζουμε τη σειρά με την οποία θα επιστρέψει το MMTx το σύνολο των εννοιών που αντιστοιχούν στην καλύτερη αντιστοίχηση. Ένα τέτοιο παράδειγμα είναι η έννοια της οντολογίας μας "glycoprotein IIb-IIIa inhibitors" που έχει ως ένα συνώνυμό της το " Platelet Glycoprotein GPIIb-IIIa Complex inhibitors". Αν η λέξη κλειδί ή φράση είναι η " glycoprotein IIb IIIa inhibitors" το MMTx μετά την επεξεργασία της, θα επιστρέψει τον συνδυασμό των εννοιών "Platelet Glycoprotein GPIIb-IIIa

Complex" και "inhibitors" ως έννοιες του UMLS που αποτελούν την καλύτερη αντιστοίχηση για τον αρχικό όρο αναζήτησης. Αν πάρουμε όμως όλες τις δυνατές μεταθέσεις των εννοιών αυτών θα έχουμε τους συνδυασμούς " Platelet Glycoprotein GPIIb-IIIa Complex inhibitors" και "inhibitors Platelet Glycoprotein GPIIb-IIIa Complex inhibitors", που ο πρώτος ταιριάζει με τον συνώνυμο όρο που υπάρχει στην οντολογία, οπότε θα έχουμε και επιτυχία στην αναζήτηση. Στο Σχήμα 7.4 φαίνεται μια σχηματική αναπαράσταση του προηγούμενου παραδείγματος.



Σχήμα 7.4 Παράδειγμα χρήσης μεταθέσεων της εξόδου του MMTx.

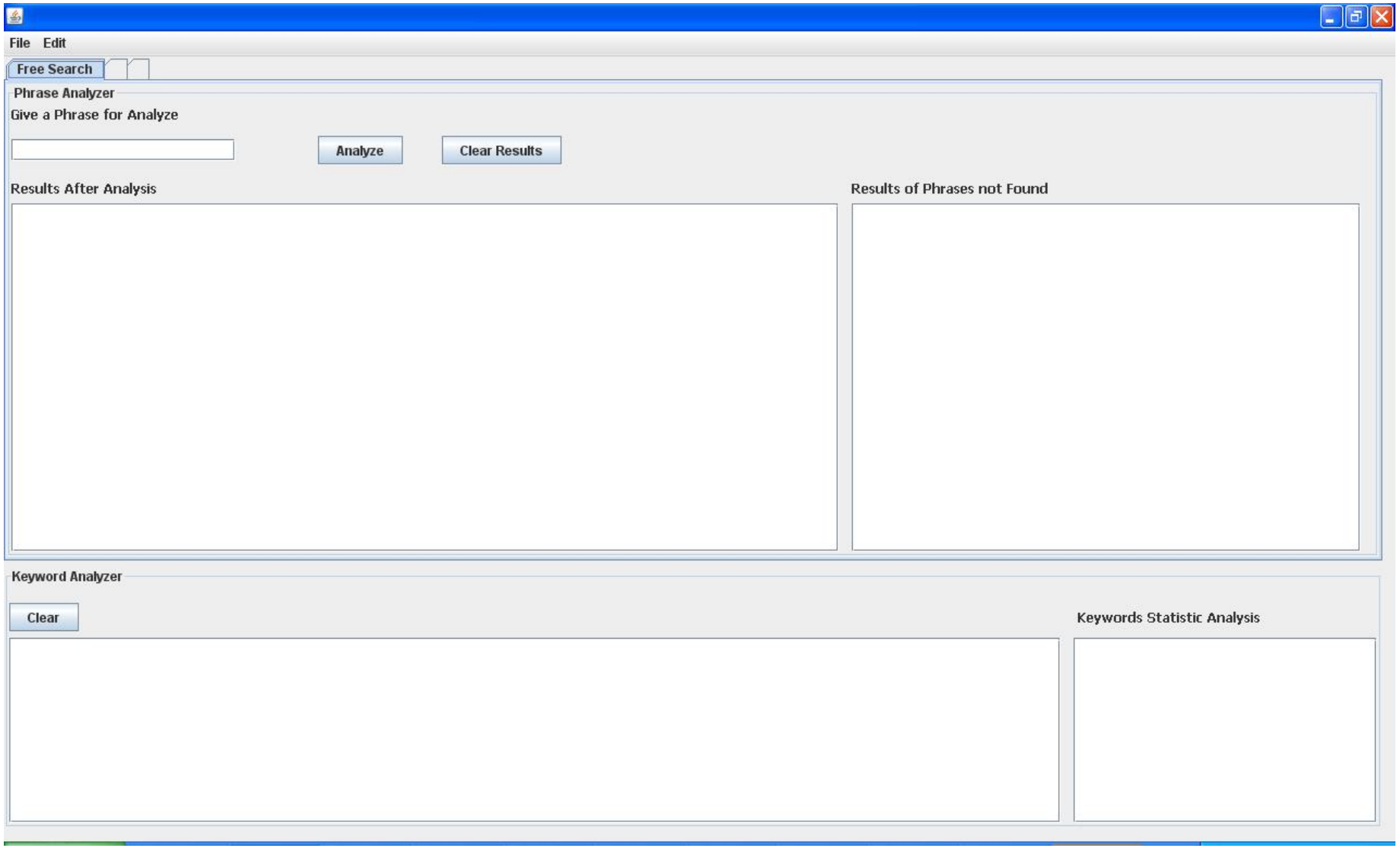
7.3. Παραδείγματα του Συστήματος

7.3.1. Το Γραφικό περιβάλλον (GUI)

Στο Σχήμα 7.5 βλέπουμε το αρχικό γραφικό περιβάλλον της εφαρμογής. Επάνω αριστερά υπάρχει το μενού "File" μέσω του οποίου μπορεί ο χρήστης να δώσει ως είσοδο τα διάφορα αρχεία που είναι απαραίτητα για να λειτουργήσει το πρόγραμμα (OWL αρχείο, αρχείο που περιέχει τα έγγραφα και τις αντίστοιχες λέξεις κλειδιά και το αρχείο με τις λέξεις σταματήματος "Stop Words"). Επίσης μέσω του μενού "File" μπορεί να κάνει αποθήκευση της οντολογίας μετά από μια λειτουργία δεικτοδότησης (document indexing) μιας λίστας εγγράφων, ώστε το OWL αρχείο να είναι σε θέση να χρησιμοποιηθεί σε μετέπειτα αναζητήσεις

Επίσης στο επάνω αριστερό παράθυρο βλέπουμε το σημείο όπου κάποιος χρήστης μπορεί να εισάγει την ερώτηση του σε φυσική γλώσσα "Give a Phrase for Analyze", και πατώντας το κουμπί "Analyze", να πάρει τα αποτελέσματα, δηλαδή τα έγγραφα που αντιστοιχεί το σύστημα στο ερώτημα, στο ακριβώς κάτω παράθυρο "Results After Analysis". Στο δεξί και επάνω παράθυρο "Results of Phrases not Found", δίνονται στοιχεία φράσεων της ερώτησης για τις οποίες δεν έχει βρεθεί κάποια αντιστοίχηση στην οντολογία.

Στο κάτω αριστερό μέρος του Σχήματος 7.5 "Keyword Analyzer" παρουσιάζονται στον χρήστη στοιχεία της αντιστοίχησης των λέξεων κλειδιών των εγγράφων προς εισαγωγή στο σύστημα, με συστατικά της οντολογίας. Τέλος στο κάτω και δεξιά παράθυρο "Keyword Statistic Analysis" εμφανίζεται το πλήθος των λέξεων κλειδιών που υπήρχαν στο αρχείο με τα έγγραφα προς δεικτοδότηση, το πλήθος των λέξεων κλειδιών που αντιστοιχήθηκαν με συστατικά της οντολογίας και το πλήθος αυτών που δεν ήταν εφικτό να πραγματοποιηθεί αντιστοιχία.



Σχήμα 7.5 Το Γραφικό Περιβάλλον της Εφαρμογής.

7.3.2. Διαδικασία Δεικτοδότησης Εγγράφων (Documents Indexing)

Στο Σχήμα 7.7 βλέπουμε τη λειτουργία της δεικτοδότησης κειμένων. Εισάγαμε στο σύστημα την οντολογία, καθώς και ένα αρχείο με ένα σύνολο των 81 εγγράφων και τις αντίστοιχες λέξεις κλειδιά τους. Τα έγγραφα διαλέχτηκαν με τέτοιο τρόπο ώστε να έχουν λέξεις κλειδιά, τέτοιες ώστε κάθε instance της οντολογίας, να έχει και ένα αντιπροσωπευτικό έγγραφο. Δηλαδή κάθε instance να έχει τουλάχιστον μια σχέση, "hasDocument" στην οντολογία, με ένα τουλάχιστον έγγραφο. Στο κάτω αριστερό μέρος του Σχήματος 7.7 βλέπουμε τις αντιστοιχίσεις που το σύστημα έχει κάνει μεταξύ συστατικών της οντολογίας (classes ή instances) και των λέξεων κλειδιών του εκάστοτε εγγράφου. Οι αντιστοιχίσεις που για το έγγραφο με ID 141 φαίνονται στο Σχήμα 7.6.

For the document with ID:Document141 And Title: Delayed Preconditioning-Mimetic Actions of Nitroglycerin in Patients Undergoing Exercise Tolerance Tests made the below keywords mappings at our ontology elements:

Ontology Element:http://www.owl-ontologies.com/unnamed.owl#exercise For Keyword:exercise
Ontology Element:http://www.owl-ontologies.com/unnamed.owl#nitroglycerin For Keyword:nitroglycerin
Ontology Element:http://www.owl-ontologies.com/unnamed.owl#myocardial_ischemia For Keyword:ischemia
Ontology Element:http://www.owl-ontologies.com/unnamed.owl#angina_pectoris For Keyword:angina
No Ontology element has mapping for the keyword:coronary disease, Please update the ontology

Σχήμα 7.6 Το Γραφικό Περιβάλλον της Εφαρμογής.

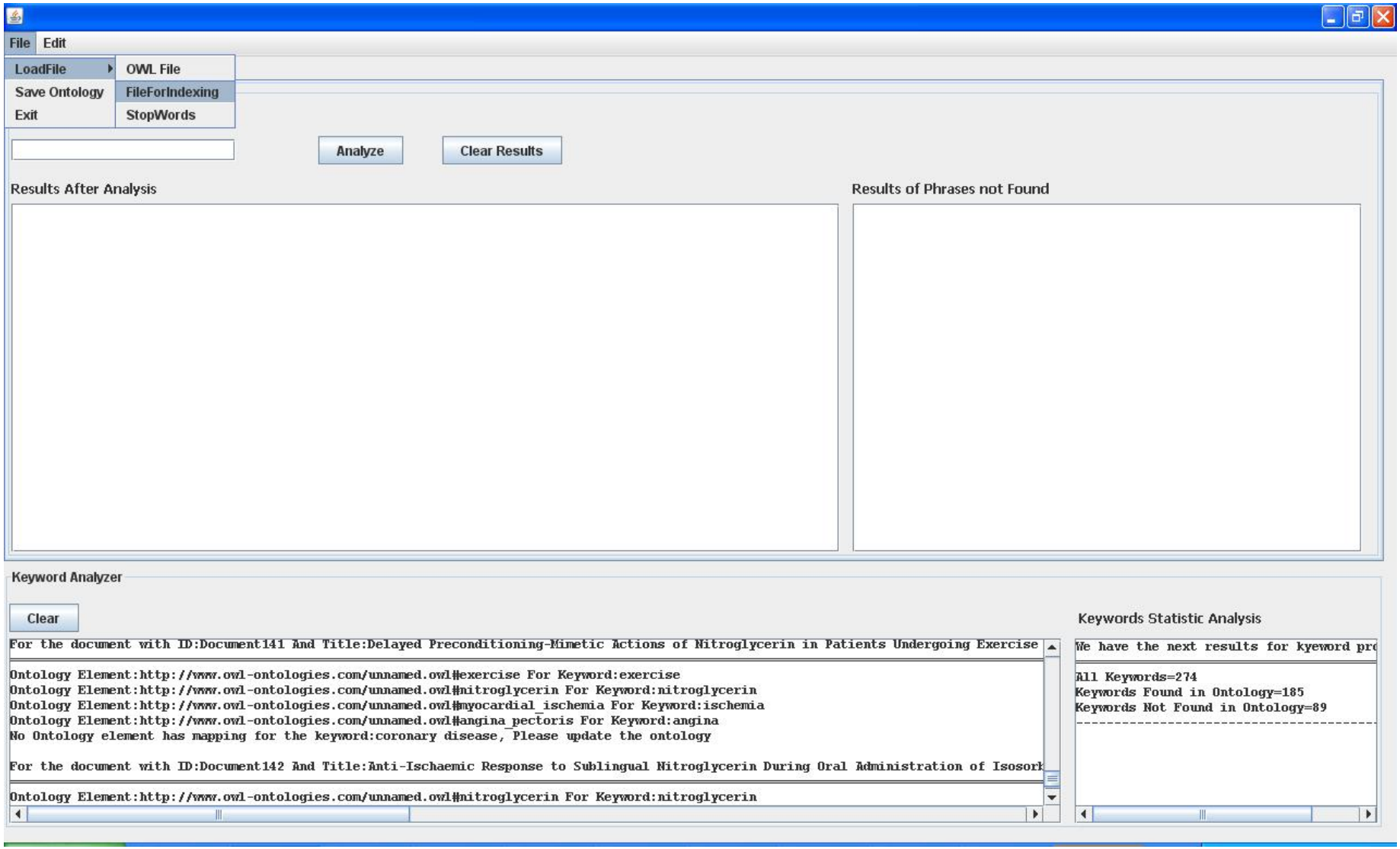
Βλέπουμε πως για την λέξη κλειδί "exercise" έγινε αντιστοίχηση με το instance "exercise" της οντολογίας. Στην περίπτωση αυτή, η λέξη κλειδί δεν θα περάσει από το MMTx αφού θα βρεθεί αυτούσια εξαρχής στην οντολογία. Η διαδικασία εδώ θα είναι η εξής: το σύστημα δημιουργεί ένα instance στην κλάση Document με όνομα "Document141", θα λάβει τη λέξη κλειδί "exercise" και μέσω του ARQ API, και χρησιμοποιώντας SPARQL, θα κάνει ερώτηση στην οντολογία για το εάν υπάρχει συστατικό στην οντολογία με όνομα ή συνώνυμό του το "exercise". Αφού βρεθεί και βεβαιωθεί πως είναι "instance" δημιουργεί μια σχέση τύπου "hasDocument" μεταξύ του "exercise" και του

“Document141”. Αυτόματα ο Pellet reasoner μέσω του μηχανισμού συμπερασμού, συμπεραίνει και δημιουργεί τη σχέση “isDocumentOf”, μεταξύ των instances “Document141” και “exercise”. Αυτό γίνεται γιατί στην οντολογία έχει δηλωθεί πως τα δύο properties, “hasDocument” και “isDocumentOf” είναι inverse properties. Όλες οι παραπάνω διαδικασίες γίνονται μέσω του Jena API.

Το ίδιο θα συμβεί και με τις υπόλοιπες λέξεις κλειδιά. Εκτός της τελευταίας “coronary disease”, για την οποία ακόμη και αν έχει υποστεί επεξεργασία μέσω του MMTx, το σύστημα δεν την έχει εντοπίσει κάπου στην οντολογία. Το MMTx αποδίδει την αντιστοιχία του “coronary disease” στην έννοια “coronary heart disease” του UMLS. Η οντολογία όμως δεν συμπεριλαμβάνει αυτό τον όρο. Συμπεριλαμβάνει όμως τον όρο “coronary artery disease”. Οπότε το σύστημα ενημερώνει τον χρήστη να προχωρήσει σε ενημέρωση της οντολογίας για να συμπεριληφθεί ο όρος, είτε σαν νέα έννοια (Class ή Instance) είτε σαν συνώνυμο μιας που υπάρχει ήδη.

Στην περίπτωση που το συστατικό της οντολογίας που θα βρεθεί να ταιριάζει με μία λέξη κλειδί είναι κλάση, υπάρχει μια διαφοροποίηση στον τρόπο λειτουργίας του προγράμματος. Αν για παράδειγμα βρεθεί το συστατικό “Pathophysiology”, τότε μέσω του Jena API θα εντοπιστούν όλα τα instances της ίδιας της κλάσης, αλλά και όλων των υποκλάσεών της (στο σημείο αυτό παίζει πάλι ρόλο ο Pellet reasoner, που συμπεραίνει μέσω της ιεραρχίας ποια είναι τα instances της κλάσης γονέα), και θα γίνει σύνδεση μεταξύ όλων των instances και του εγγράφου, μέσω των σχέσεων που προαναφέρθηκαν

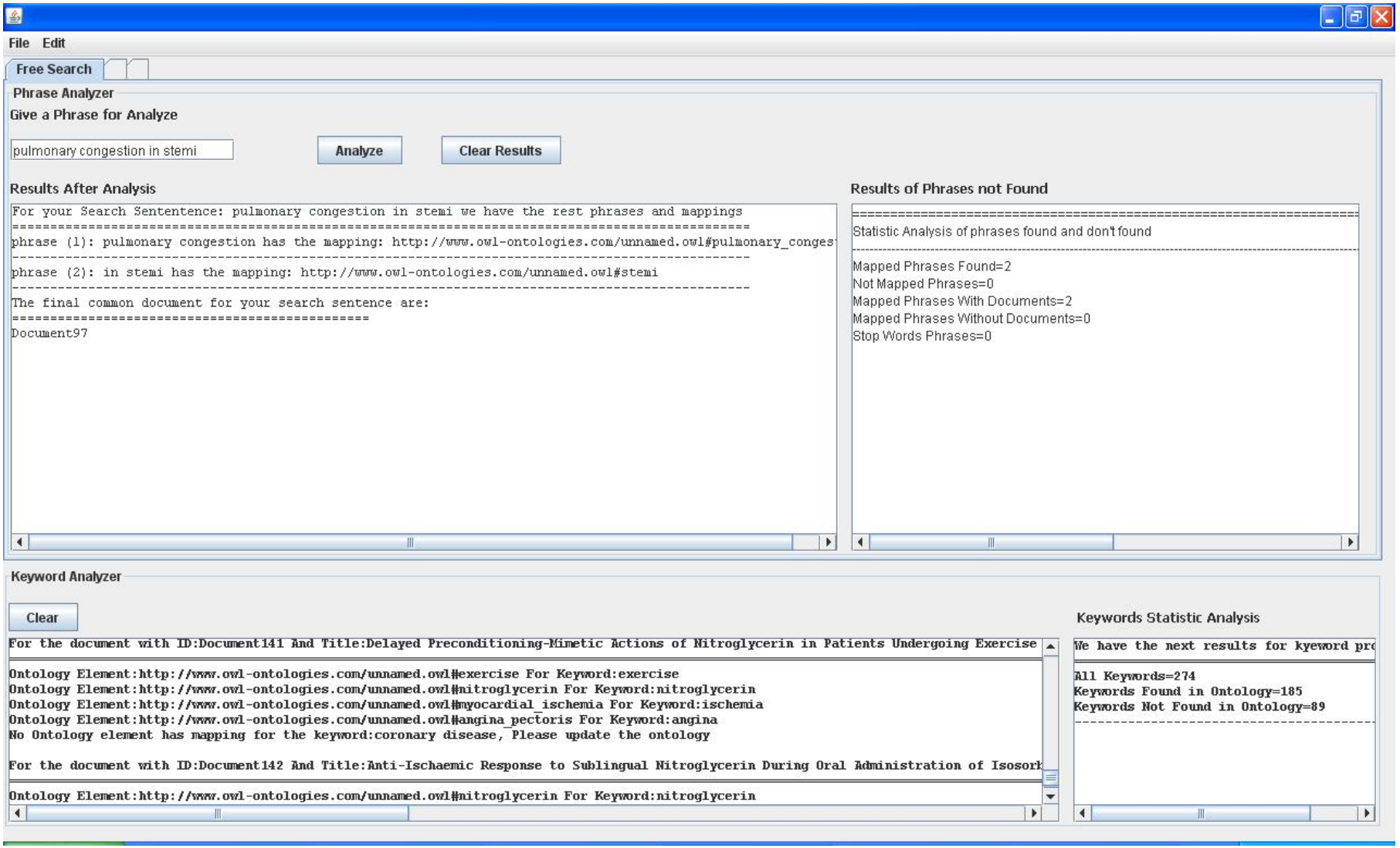
Στο κάτω δεξιό τμήμα του Σχήματος 7.7.το σύστημα ενημερώνει τον χρήστη πως από σύνολο 274 λέξεων κλειδιών που υπήρχαν στο αρχείο εγγράφων προς δεικτοδότηση, βρέθηκε αντιστοιχία με συστατικά της οντολογίας για τα 185 από αυτά, ενώ δεν βρέθηκε αντιστοιχία για τα 89.



Σχήμα 7.7 Η Διαδικασία Δεικτοδότησης Κειμένων (Documents Indexing).

7.3.3. Διαδικασία Ανάκτησης Κειμένων

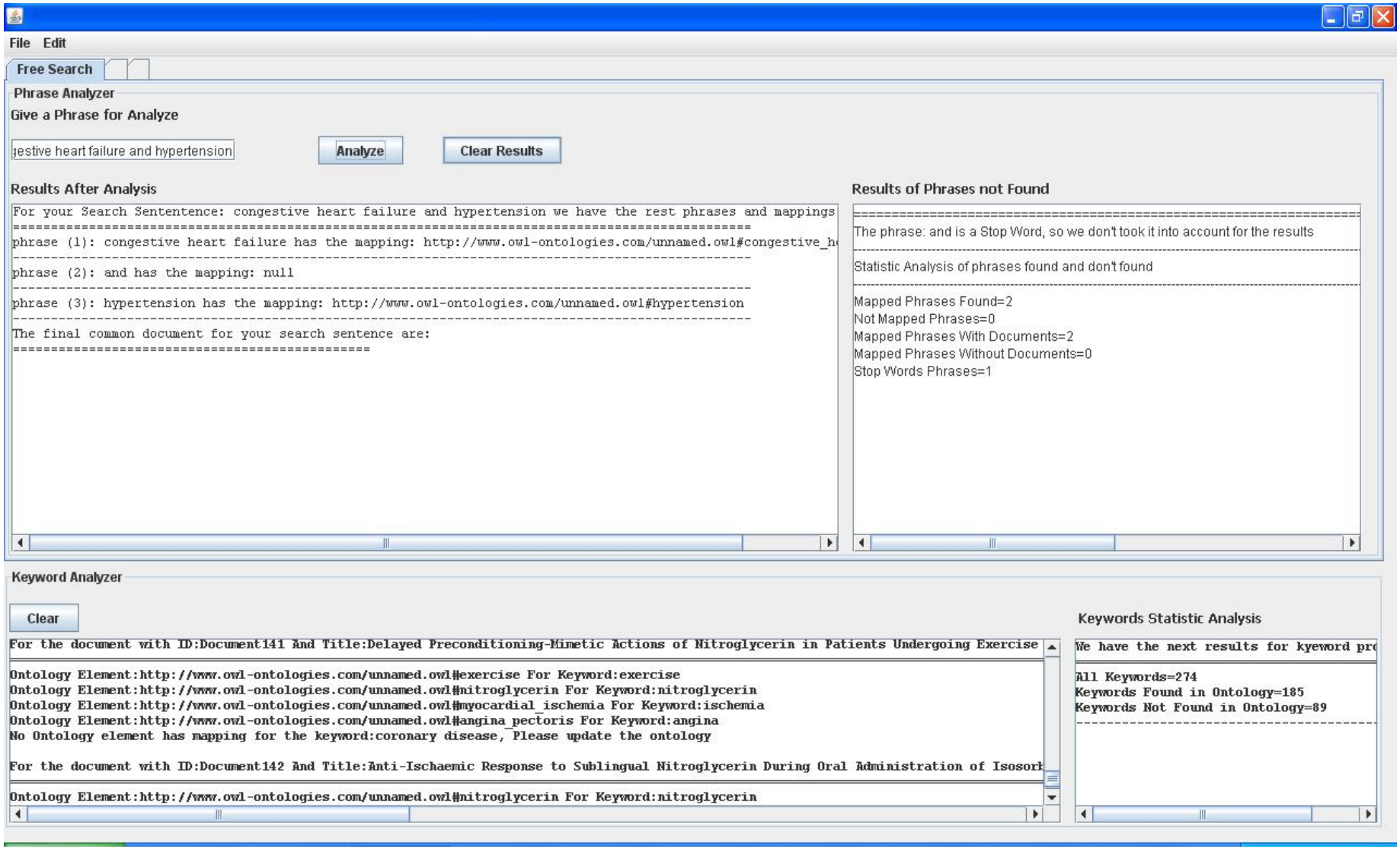
Το σύστημα στην περίπτωση ανάκτησης κειμένων, δέχεται από τον χρήστη μια ερώτηση σε μορφή φυσικής γλώσσας. Η Ερώτηση αυτή δέχεται επεξεργασία από το MMTx ώστε αυτό με τη σειρά του να εξάγει τις συστατικές φράσεις της ερώτησης. Γίνεται ένας αρχικός έλεγχος της οντολογίας (μέσω του ARQ API και της γλώσσας SPARQL), για το εάν υπάρχει αυτούσια εξαρχής κάποια αντιστοίχηση των φράσεων στην οντολογία, εάν για κάποια φράση έχουμε επιτυχία σε αυτό το στάδιο, τότε σταματά η παρακάτω επεξεργασία της. Αυτό γίνεται γιατί όπως περιγράφηκε και για τις λέξεις κλειδιά, υπάρχουν όροι της οντολογίας που εισήλθαν από τον ιατρό, αλλά δεν υπήρχαν αυτούσιοι στο UMLS, οπότε και να υποστούν επεξεργασία για την αντιστοίχηση τους σε UMLS έννοιες, δεν θα έχουμε επιτυχία. Για τις υπόλοιπες φράσεις το MMTx επιστρέφει την καλύτερη αντιστοίχησή τους σε έννοιες του UMLS. Η αντιστοίχηση αυτή μπορεί να είναι είτε μία έννοια είτε ένα σύνολο εννοιών. Στην περίπτωση που η καλύτερη αντιστοίχηση είναι μια έννοια, τότε γίνεται αναζήτηση αντιστοίχησής της στην οντολογία, αντίθετα αν πρόκειται για σύνολο εννοιών γίνεται αναζήτηση για αντιστοίχηση στην οντολογία, μιας από τις δυνατές μεταθέσεις του συνόλου των εννοιών. Το στάδιο αυτό κάνει χρήση για ακόμη μια φορά του ARQ API και της SPARQL γλώσσας επερωτήσεων οντολογιών. Η χρήση των δυνατών μεταθέσεων του συνόλου εννοιών του UMLS, γίνεται λόγω του ότι ένας τέτοιος όρος μπορεί να έχει εισαχθεί στην οντολογία από τον ειδικό του πεδίου. Πλέον έχουμε στη διάθεση μας ένα σύνολο από έννοιες της οντολογίας μας, οι οποίες αντιπροσωπεύουν το ερώτημα του χρήστη. Φράσεις που δεν βρέθηκαν να αντιστοιχούν σε όρους της οντολογίας δεν λαμβάνονται υπόψη και ενημερώνεται ο χρήστης. Επιπλέον στο σύστημα έχει δοθεί ως είσοδος μια λίστα από λέξεις της αγγλικής που δεν έχουν να κάνουν με το πεδίο ενδιαφέροντος, αλλά χρησιμοποιούνται ευρέως (Stop Words). Αν μια φράση αντιστοιχεί σε μια από τις λέξεις αυτές, τότε παραβλέπεται από το σύστημα ώστε να μην διαδραματίσει ρόλο στα αποτελέσματα του ερωτήματος. Τελικά το σύστημα αναζητά τα έγγραφα της οντολογίας, που έχουν κάποια σχέση με τις έννοιες που έχουμε στη διάθεσή μας. Ουσιαστικά διατρέχει για κάθε έννοια τη σχέση “hasDocument”, της οντολογίας και συλλέγει τα instances της κλάσης “Document” που είναι συνδεμένα με αυτή. Το σύνολο των εγγράφων που δίνονται ως απάντηση στον χρήστη, είναι το σύνολο των κοινών εγγράφων, των εννοιών της οντολογίας που αντιπροσωπεύουν το ερώτημα. Σε όλα τα παραπάνω στάδια σε ότι έχει να κάνει με τη διαχείριση της οντολογίας, γίνεται χρήση του Jena API.



Σχήμα 7.8 Αναζήτηση για Έγγραφα Σχετικά με “Pulmonary Congestion in stemi”.

Στο Σχήμα 7.8 βλέπουμε ένα παράδειγμα αναζήτησης εγγράφων. Ο χρήστης τοποθετεί το ερώτημά του “Pulmonary Congestion in stemi”, και πατώντας το κουμπί “Analyze” αναμένει για αποτελέσματα εγγράφων σχετικών με το ερώτημά του. Βλέπουμε πως το MMTx έχει διαχωρίσει το ερώτημα σε δύο φράσεις την “ pulmonary congestion” και την “ in stemi”. Μέσω του ARQ API και της SPARQL, αναζητάμε στην οντολογία να δούμε αν υπάρχουν οι φράσεις αυτούσιες. Η πρώτη υπάρχει οπότε κρατάτε το instance που βρέθηκε “pulmonary congestion”. Η δεύτερη δεν υπάρχει οπότε συνεχίζει η περαιτέρω επεξεργασία για αυτή. Το MMTx λαμβάνει η φράση “in stemi” και επιστρέφει την πιο αντιπροσωπευτική έννοια του UMLS που μπορεί να της αντιστοιχηθεί. Η έννοια του UMLS που επιστρέφει το MMTx είναι η “stemi”. Γίνεται και πάλι αναζήτηση της οντολογίας, μέσω SPARQL ερωτήματος, για την ύπαρξη κάποιου συστατικού της που να έχει ως ετικέτα ή ως συνώνυμο του την έννοια “stemi”. Το SPARQL ερώτημα επιστρέφει την έννοια “stemi” από την οντολογία. Στη συνέχεια μέσω του Jena API, λαμβάνουμε την έννοια αυτή και ελέγχουμε αν είναι class ή instance στην οντολογία. Στην περίπτωσή μας είναι instance, οπότε και το αντιστοιχούμε με τη φράση “in stemi” του ερωτήματος. Αφού έχει τελειώσει η επεξεργασία όλων των φράσεων και έχουν βρεθεί αντιστοιχίσεις τους με την οντολογία, παίρνουμε όλα τα instances της οντολογίας που αντιστοιχήθηκαν στις φράσεις, με το Jena API και λαμβάνουμε το κοινό σύνολο των instances, της κλάσης Document της οντολογίας, που συνδέονται μέσω της σχέσης “hasDocument” με τα instances της αντιστοίχησης. Αυτό το σύνολο είναι και τα κοινά έγγραφα που επιστρέφονται ως απάντηση στον χρήστη. Στο παράδειγμά μας το κοινό αυτό σύνολο αποτελείται από ένα έγγραφο, το Document97.

Στο επάνω δεξιό τμήμα της εφαρμογής, βλέπουμε πως επιστρέφονται και κάποια ενημερωτικά μηνύματα στον χρήστη. Στο συγκεκριμένο παράδειγμα ενημερώνει πως από τις δύο συστατικές φράσεις του ερωτήματος, έγινε αντιστοίχηση με συστατικά της οντολογίας και για τις δύο. Επίσης αναφέρει πως και οι δύο φράσεις αντιστοιχήθηκαν σε instances της οντολογίας που είχαν σύνδεση με κάποια έγγραφα και πως δεν εμπεριέχεται κάποια λέξη σταματήματος (Stop Words) στις φράσεις που εντόπισε το σύστημα.



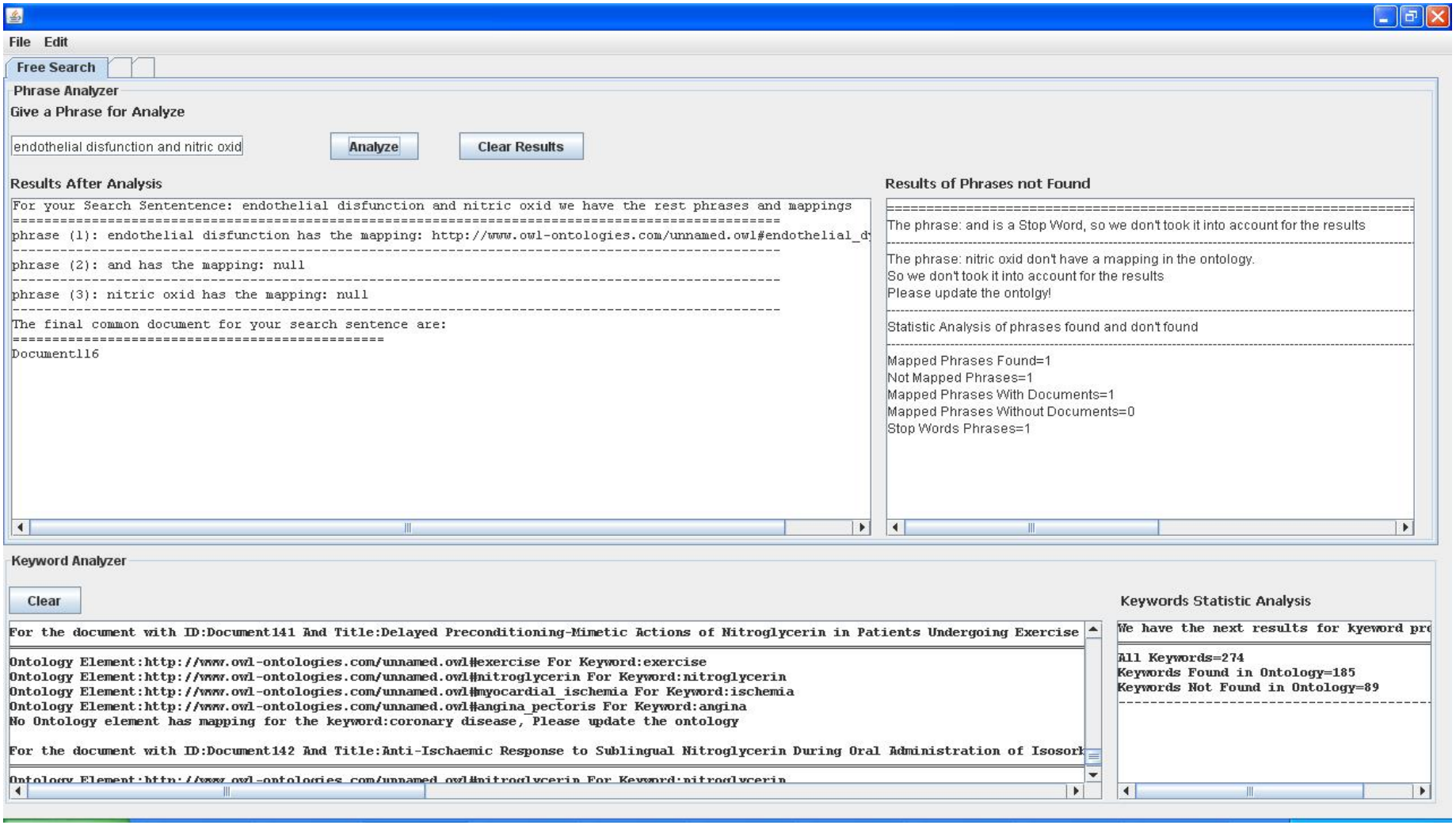
Σχήμα 7.9 Αναζήτηση για Έγγραφα Σχετικά με “Congestive Heart Failure and Hypertension”.

Στο Σχήμα 7.9 βλέπουμε την αναζήτηση που γίνεται με βάση το ερώτημα “Congestive Heart Failure and Hypertension”. Εδώ γίνεται η ίδια διαδικασία όπως και στο προηγούμενο παράδειγμα, με τη διαφορά πως η μια συστατική φράση του ερωτήματος είναι η “and” η οποία εμπεριέχεται στη λίστα των Stop Words που δόθηκες ως είσοδος στο σύστημα, οπότε και η φράση αυτή δεν λαμβάνει χώρα στην αποτίμηση του ερωτήματος. Επίσης παρατηρούμε πως δεν επιστρέφονται κάποια έγγραφα στον χρήστη. Αυτό συμβαίνει γιατί ναι μεν και τα δυο instances που βρέθηκαν στην οντολογία συνδέονται με κάποια έγγραφα, πράγμα που φαίνεται και στα ενημερωτικά μηνύματα δεξιά της οθόνης, αλλά τα δύο instances δεν έχουν κοινά έγγραφα μεταξύ τους.

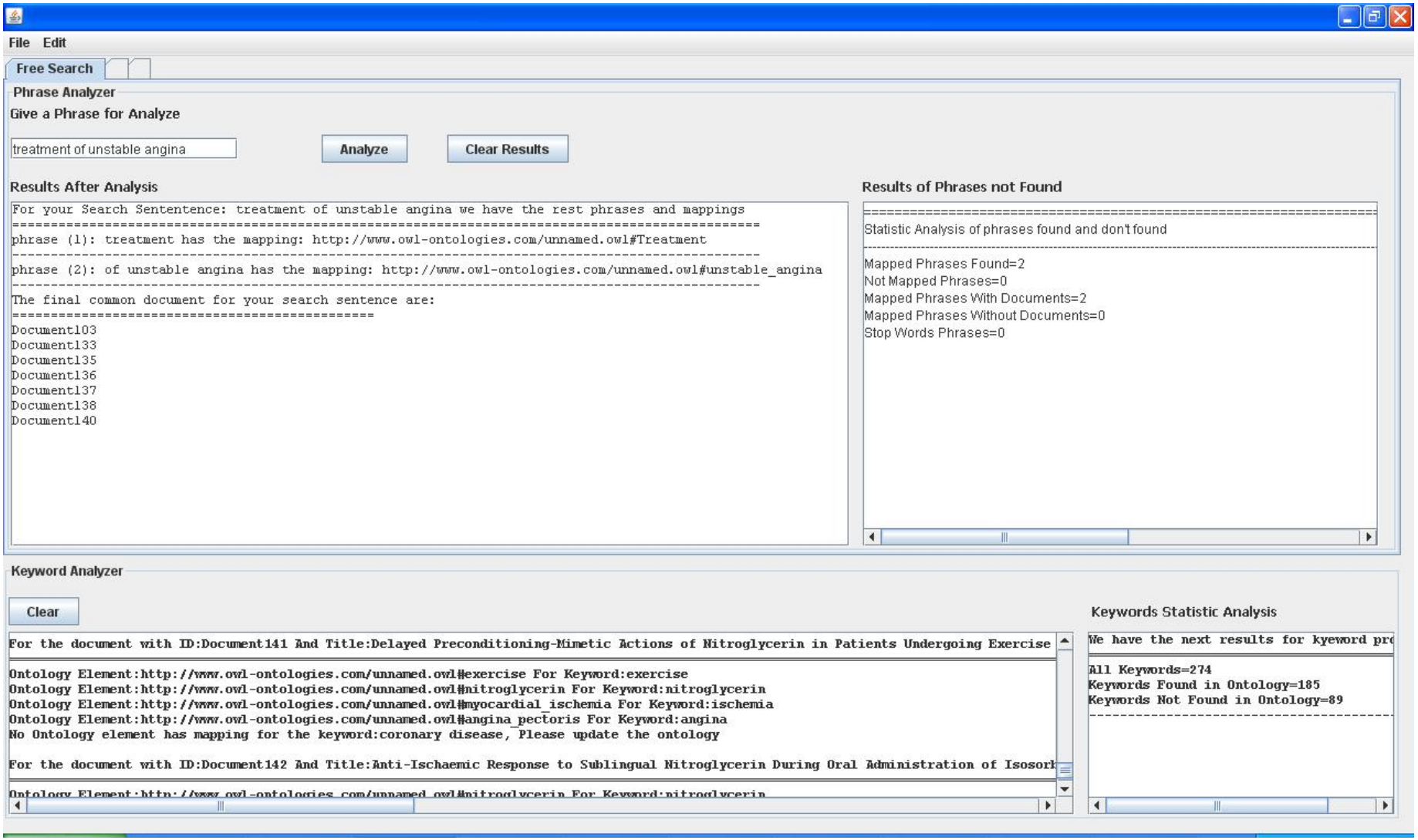
Στο Σχήμα 7.10 βλέπουμε ένα ακόμη παράδειγμα αναζήτησης. Εδώ το ερώτημα είναι το “Endothelial Disfunction and Nitric Oxid”. Η διαφορά του παραδείγματος αυτού από τα προηγούμενα είναι πως από τις δύο φράσεις “Endothelial Disfunction” και “Nitric Oxid” βρέθηκε μόνο για τη μια αντιστοίχηση όρου στην οντολογία. Ο όρος “Nitric Oxid” δεν αντιστοιχίζεται στην οντολογία μας. Το σύστημα ενημερώνει (επάνω δεξιό παράθυρο) τον χρήστη πως δεν θα λάβει υπόψη του τη φράση που δεν βρέθηκε στα αποτελέσματα που θα επιστρέψει, όπως επίσης πως πρέπει να ενημερωθεί η οντολογία για να μπορέσει να υποστηρίξει και τον όρο που δεν βρέθηκε. Έτσι το έγγραφο “Document116” αναφέρεται μόνο στη φράση “Endothelial Disfunction” του ερωτήματος.

Στο Σχήμα 7.11 υπάρχει μια διαφοροποίηση σε σχέση με όλα τα υπόλοιπα παραδείγματα. Το ερώτημα στο παράδειγμα αυτό είναι το “Treatment of Unstable Angina”. Η φράση “Treatment” αντιστοιχεί σε κλάση της οντολογίας και όχι σε instance. Εδώ αναδεικνύεται μια σημασιολογική χρήση της οντολογίας. Η κλάση Treatment έχει τα δικά της instances καθώς και ένα σύνολο από υποκλάσεις με τα δικά τους instances, που επίσης οι υποκλάσεις έχουν άλλες υποκλάσεις κ.ο.κ. Στο συγκεκριμένο παράδειγμα το σύστημα θα συγκεντρώσει τα έγγραφα που είναι συνδεδεμένα με το σύνολο των instances της κλάσης Treatment καθώς και όλων των υποκλάσεων της, και θα τα συγκρίνει με το σύνολο των εγγράφων που είναι συνδεδεμένα με το instance “Unstable Angina”. Στον χρήστη θα επιστρέψει τα κοινά έγγραφα των δύο συνόλων. Ουσιαστικά επιστρέφει έγγραφα που σχετίζονται με το “unstable angina” και τη θεραπεία του, όποια και αν είναι αυτή.

.



Σχήμα 7.10 Αναζήτηση Εγγράφων για το Ερώτημα “Endothelial Disfunction and Nitric Oxid”.



Σχήμα 7.11 Αναζήτηση Εγγράφων για το Ερώτημα “Treatment of Unstable Angina”.

ΚΕΦΑΛΑΙΟ 8. ΣΥΜΠΕΡΑΣΜΑΤΑ-ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Κατά τη διάρκεια αυτής της μεταπτυχιακής εργασίας μελετήθηκε το πεδίο των οντολογιών και η χρήση τους στην πληροφορική και ιδιαίτερα σε συστήματα ιατρικής πληροφορίας. Οι οντολογίες είναι το μέσο που προσφέρει σε τέτοια συστήματα, την ικανότητα οργάνωσης της πληροφορίας που είναι διάχυτη στο εκάστοτε πεδίο ενδιαφέροντος, καθώς και την ικανότητα διαμοιρασμού της. Μελετήθηκε το θεωρητικό υπόβαθρο των οντολογιών, καθώς και εργαλεία και τεχνικές για την ανάπτυξή τους και τη χρήση τους σε συστήματα που χρησιμοποιούνται στην κλινική πράξη. Μέσω της οντολογίας στο πεδίο των καρδιαγγειακών νοσημάτων περιγράφηκε και οργανώθηκε η πληροφορία που είναι συσσωρευμένη στο πεδίο αυτό. Και μέσω του βασισμένου σε οντολογίες συστήματος ανάκτησης κειμένων, έγινε εκμετάλλευση της οντολογίας, ώστε να χρησιμοποιηθούν σημασιολογικά κριτήρια στην ανάκτηση εγγράφων. Με τον τρόπο αυτό επιτεύχθηκε και η οργάνωση της πληροφορίας, αλλά και ο διαμοιρασμός της, από τη στιγμή που ένας υπολογιστής έφτασε στη θέση να την κατανοεί.

Η οντολογία δημιουργήθηκε από την αρχή, με τη βοήθεια ενός ειδικού στο πεδίο ενδιαφέροντος, ενός ιατρού. Έγινε η προσπάθεια να αντικατοπτριστεί η πληροφορία του πεδίου με όσο το δυνατόν καλύτερο και πρακτικό τρόπο, αλλά και με όσο το δυνατόν ευρέως αποδεκτούς όρους. Αυτό το τελευταίο επιτεύχθηκε μέσω της χρήσης της ευρέως αποδεκτής οντολογίας στον τομέα της ιατρικής UMLS. Επίσης δημιουργήθηκε με τέτοιο τρόπο, ώστε να μην είναι οργανωμένη η πληροφορία που συμπεριλαμβάνει σύμφωνα με τις απαιτήσεις ενός συστήματος ανάκτησης κειμένων, αλλά να είναι σε θέση να χρησιμοποιηθεί και σε συστήματα που έχουν εφαρμογή σε διαφορετικούς τομείς

Με τη χρήση του φραστικού αναλυτή MMTx , επιτεύχθηκε η σύγκριση μεταξύ κοινών πραγμάτων. Από τη μια η οντολογία, που χρησιμοποιεί έννοιες του UMLS και από την άλλη η αντιστοίχηση των λέξεων κλειδιών του κάθε κειμένου και των συστατικών φράσεων του ερωτήματος του χρήστη σε έννοιες του UMLS.

Βέβαια γενικά μια οντολογία από τη στιγμή που έχει ολοκληρωθεί η δομή της, και αρχίσει η χρήσης της σε κάποιο πεδίο, έχει ανάγκη από συνεχόμενη ενημέρωσή της. Αυτό είναι πολύ πιο απαραίτητο σε τομής όπως η ιατρική, όπου συνέχεια η πληροφορία αυξάνεται, ή μεταβάλλεται. Έτσι και η οντολογία που δημιουργήθηκε έχει την άμεση ανάγκη από εμπλουτισμό και όχι μόνο από έναν άνθρωπο-ειδικό, αλλά από ένα σύνολο ειδικών του πεδίου, ώστε να συγκεντρωθεί όσο το δυνατόν περισσότερη γνώση σε αυτή. Ο κάθε άνθρωπος μπορεί να κάνει την αναζήτησή του σε μια βάση από έγγραφα με πλήθος διαφορετικών τρόπων, έτσι η οντολογία έχει ανάγκη να ενσωματώσει όσο το δυνατόν περισσότερους συνώνυμους όρους για τις έννοιες που συμπεριλαμβάνει.

Το σύστημα ανάκτησης εγγράφων, δεικτοδοτεί τα έγγραφα σε κόμβους της οντολογίας, βασιζόμενο στις λέξεις κλειδιά που δίνονται για το κάθε έγγραφο. Αυτό σημαίνει πρακτικά πως ο χρήστης που δίνει τις λέξεις κλειδιά για κάποιο κείμενο πρέπει να είναι ιδιαίτερα προσεκτικός ώστε οι λέξεις αυτές να αντικατοπτρίζουν όσο το δυνατόν καλύτερα το περιεχόμενο του κειμένου. Για μια πιο αυτόματη δεικτοδότηση κειμένων σε συστατικά της οντολογίας, υπάρχει η ανάγκη ενσωμάτωσης ενός υποσυστήματος, που θα ανακτά αυτόματα από το περιεχόμενο των εγγράφων τις καταλληλότερες λέξεις που αντικατοπτρίζουν και το περιεχόμενό του.

Επίσης εμείς χρησιμοποιήσαμε το MMTx για την επεξεργασία των ερωτημάτων που δίνονται σε φυσική γλώσσα, μιας που έχει δημιουργηθεί από τον ίδιο οργανισμό που έχει δημιουργήσει και το UMLS. Ίσως αν κάποιος αναπτύξει κάποιο λεξικό, βασιζόμενος στην ίδια την οντολογία του πεδίου των καρδιαγγειακών νοσημάτων και κάνει χρήση αυτού με κάποιον διαφορετικό φραστικό αναλυτή, να μπορέσει να αντιστοιχήσει με καλύτερο τρόπο τις φράσεις που αποτελούν το ερώτημα ενός χρήστη σε όρους της οντολογίας.

ΑΝΑΦΟΡΕΣ

[1] F. Pincirolia, D. Pisanelli, “The unexpected high practical value of medical Ontologies”, Computers in Biology and Medicine, Vol. 36 (7-8), pp. 667-673, jul-Aug 2006.

[2] T. Gruber, “A translation approach to portable ontologies”, Knowledge Acquisition, Vol. 5 (2), pp. 199–220, June 1993.

[3] N. Guarino, “Formal ontology and information systems”, Formal Ontology in Information Systems, FOIS’ 1998, pp. 3-15, June 1998.

[4] http://www.jfsowa.com/ontology/ontoshar.htm.

[5] http://ontology.buffalo.edu/ontology(PIC).pdf

[6] R. Neches, R. Fikes, T. Finin, T. Gruber, T. Senator, W. Swartout, “Enabling technology for knowledge sharing”, AI Magazine, Vol. 12(3), pp.36-56, Fall 1991.

[7] T. Gruber. “Towards principles for the design of ontologies used for knowledge sharing”, International Journal of Human-Computer Studies, Vol. 43(5-6):pp. 907-928, November 1995.

[8] J. Arpirez, A. Gomez-Perez, A. Lozano, H. Pinto, “(ONTO)2 Agent: An ontology-based WWW broker to select ontologies”, Workshop on Applications of Ontologies and Problem-Solving Methods, 13th ECAI’98, pp. 16-24, January 1998.

[9] M. Fernandez-Lopez, A. Gomez-Perez, N. Juristo, “METHODOLOGY: From Ontological Art Towards Ontological Engineering”, Spring Symposium on Ontological Engineering of AAAI97, Stanford University, pp. 33-40, March 1997.

[10] IEEE Standard for Developing Software Life Cycle Processes, IEEE Computer Society, New York, IEEE std 1074-1995

[11] J. Kietz, A. Maedche, R. Volz, “A method for Semi-Automatic Ontology Acquisition from a Corporate Intranet”, WS on Ontologies and Text, co-located with EKAW'2000, October 2000.

[12] A. Gomez-Perez, “Some Ideas and Examples to Evaluate Ontologies”, 11th Conference on Artificial Intelligence for Applications, pp. 299-305, February 1995.

[13] T. Declerc, H. Uszkoreit, “State of the art on multilinguality for ontologies, annotation services and user interfaces”, Deliverable Espeonto Project IST- 2001-34373, January 2003

[14] A. Gomez-Perez, M. Fernandez-Lopez, O. Corcho, “Ontological Engineering”, Springer-Verlag New York, Inc., Secaucus, NJ, 2003.

[15] A. Gomez-Perez, M. Rojas, “Ontological Reengineering and Reuse”, Springer-Verlag London, UK, pp. 139 – 156, 1999.

[16] D. Lenat, R. Guha, “Building Large Knowledge-based Systems: Representation and Inference in the Cyc Project”, Addison-Wesley Longman Publishing Co, Boston, 1989.

[17] M. Uschold, M. King, “Towards a Methodology for Building Ontologies”, IJCAI’95 Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Canada, pp. 6.1-6.10, 1995.

[18] M. Gruninger, M. Fox, “Methodology for the design and evaluation of ontologies”, IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing, Montreal, Canada, pp. 6.1-6.10, 1995.

[19] A. Bernaras, I. Laresgoiti, J. Corera, “Building and reusing ontologies for electrical network applications”, European Conference on Artificial Intelligence (ECAI’96), Budapest, Hungary, John Wiley & Sons, Chichester, United Kingdom, pp. 298-302, 1996.

[20] B. Swartout, P. Ramesh, K. Knight, T. Russ, “Toward Distributed Use of Large-Scale Ontologies”, AAAI’97 Spring Symposium on Ontological Engineering, Stanford University, California, pp. 138-148, 1997.

[21] M. Fernandez-Lopez, A. Gomez-Perez, A. Pazos, J. Pazos, “Building a Chemical Ontology Using Methodology and the Ontology Design Environment”, IEEE Intelligent Systems, Vol. 4(1), pp. 37-46, January-February 1999.

[22] S. Staab, H. Schnurr, R. Studer, Y. Sure, “Knowledge Processes and Ontologies”, IEEE Intelligent Systems, Vol. 16(1), pp. 26-34, January-February 2001.

[23] http://www.opengalen.org/sources/sources.html

[24] C. Eccher, B. Purin, D. Pisanelli, M. Battaglia, “Ontologies supporting continuity of care: The case of heart failure”, Computers in Biology and Medicine, Vol. 36(7-8), pp. 789-801, July-August 2006.

[25] L. Bird, A. Goodchild, Z. Tun, “Experiences with a two-level modeling approach to electronic health record”, Journal of Research and Practice in Information Technology, Vol. 35 (2), May 2003.

[26] www.openehr.org

[27] B. Orgun, J.Vu, “HL7 ontology and mobile agents for interoperability in heterogeneous medical information systems”, Computers in Biology and Medicine, Vol. 36(7), pp. 817-836, July-August 2006.

[28] P. Ganguly, P. Ray, N. Parameswaran, “Semantic Interoperability in Telemedicine through Ontology-Driven Services”, Telemedicine and e-Health., Vol. 11(3), pp. 405-412, June 2005.

[29] M . Joubert, S . Aymard, D . Fieschi, F. Volot, P. Staccini, J. Robert, M. Fieschi, “ARIANE: integration of information databases within a hospital intranet”, International Journal of Medical Informatics, Vol.49(3), pp. 297 – 309, May 1998.

[30] J. Köhler, S. Philippi and M. Lange, “SEMEDA: ontology based semantic integration of biological databases”, Bioinformatics, Vol. 19, pp. 2420–2427, December 2003.

[31] D. Pérez-Rey, V. Maojo, M. García-Remesal, R. Alonso-Calvo, H. Billhardt, F. Martin-Sánchez, A. Sousa, “ONTOFUSION: Ontology-based integration of genomic and clinical databases”, Computers in Biology and Medicine, Vol. 36(7- 8), pp. 712 – 730, July-August 2006.

[32] G. Abasolo, M. Gomez, “MELISA. An ontology-based agent for information retrieval in medicine”, ECDL 2000 Workshop on the Semantic Web, Lisbona, Portugal, September 2000.

[33] J. Lin and D. Demner-Fushman, “The role of knowledge in conceptual retrieval: A study in the domain of clinical medicine”, In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 99 – 106, 2006

[34] H. Muller, E. Kenny, and P. Sternberg,. “Textpresso: An Ontology-Based Information Retrieval and Extraction System for Biological Literature”, PLoS Biology, Vol. 2(11), pp. 1984–1998, November 2004

[35] J. Paralic, I. Kostial, “Ontology-based Information Retrieval”. Information and Intelligent Systems, Croatia, pp. 23-28, 2003.

[36] A. Hliaoutakis, G. Varelas, E. Petrakis, E. Milios, “MedSearch: A Retrieval System for Medical Information Based on Semantic Similarity”, Research and Advanced Technology for Digital Libraries, Springer Berlin / Heidelberg, pp. 512-515, September 2006.

[37] G. Varelas, E. Voutsakis, P. Raftopoulou, E. Petrakis, E. Milios, “Semantic Similarity Methods in WordNet and their Application to Information Retrieval on the Web”, In: 7th ACM Intern. Workshop on Web Information and Data Management (WIDM 2005), Bremen, Germany, pp. 10–16, 2005.

[38] A. Valarakos, V. Karkaletsis, D. Alexopoulou, E. Papadimitriou, C. Spyropoulos, G. Vouros, “Building an allergens ontology and maintaining it using machine learning techniques”, Computers in Biology and Medicine, Vol. 36(10), pp. 1155-1184, October 2006.

[39] T. Bittner, “Axioms for parthood and containment relations in bio-ontologies”, in: Proceedings of the Workshop on Formal Biomedical Knowledge Representation (KR-MED), pp. 4–11, 2004.

[40] D. Benslimane, A. Arara, K. Yetongnon, F. Gargouri, H. Abdallah, “Two approaches for ontologies building: From-scratch and From existing data sources”, International Conference on Information Systems and Engineering, ISE, Montreal, Canada, July 2003.

[41] A. Baneyx, J. Charlet, M. Jaulent, “Building an ontology of pulmonary diseases with natural language processing tools using textual corpora”, International Journal of Medical Informatics, Vol. 76(2-3), pp. 208-215, February-March 2007.

[42] J. Bouaud, B. Bachimont, J. Charlet, P. Zweigenbaum, “Methodological principles for structuring an ontology”, in: Proc. IJCAI-95—Workshop on Basic Ontological Issues in Knowledge Sharing, pp. 95–148, 1995.

[43] B. Bachimont, A. Isaac, R. Troncy, “Semantic commitment for designing ontologies: a proposal”, in: A. Gomez-P´erez, V. Richard Benjamins (Eds.), Proc. EKAW, 2002.

[44] B. Habert, E. Naulleau, A. Nazarenko, “Symbolic word clustering for medium-size corpora”, in: J.-I. Tsujii (Ed.), Proc. COLING, pp. 490–495, 1996.

[45] M. Hearst, “Automatic acquisition of hyponyms from large text corpora”, in: A. Zampolli (Ed.), Proc. COLING, pp. 539–545, 1992.

[46] S. Le Moigno, J. Charlet, D. Bourigault, P. Degoulet, M. Jaulent, “Terminology extraction from text to build an ontology in surgical intensive care”, Proc. AMIA Symp., pp. 9–13, 2002.

[47] V. Malais, P. Zweigenbaum, B. Bachimont, “Mining defining contexts to help structuring differential ontologies”, Terminology, Vol. 11(1), pp. 21–53, 2005.
[48] C. Friedman, L. Shagina, Y. Lussier, G. Hripcsak, “Automated encoding of clinical documents based on natural language processing”, J. Am. Med. Inf. Assoc., Vol. 11(5), pp. 392–402, 2004.

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

[50] http://webode.dia.fi.upm.es/WebODEWeb/index.html

[51] http://www.ontoknowledge.org/tools/ontoedit.shtml

[52] http://kaon.semanticweb.org/

[53[http://www-ksl.stanford.edu/knowledge-sharing/kif/

[54] http://ksl.stanford.edu/software/ontolingua/

[55]http://www-ksl.stanford.edu/knowledge-sharing/ontologies/html/frame-ontology/index.html

[56] http://www.isi.edu/isd/LOOM/

[57] http://flora.sourceforge.net/aboutFlogic.php

[58] http://www.ai.sri.com/~okbc/

[59] http://www.cs.umd.edu/projects/plus/SHOE/

[60] http://www.ai.sri.com/pkarp/xol/

[61] http://www.w3.org/RDF/

[62] http://www.w3.org/TR/rdf-schema/

[63] http://www.ontoknowledge.org/oil/

[64] http://www.w3.org/TR/daml+oil-reference

[65] http://www.w3.org/TR/owl-features/

[66] N. Guarino, "Formal Ontology in Information Systems", In Proceedings of FOIS'98, Trento, Italy, IOS Press, Amsterdam, 1998.

[67] D. Fensel, "Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce", Second Edition, Springer – Verlag, Berlin, Heidelberg, 2004.

[68] I. Jurisica, J. Mylopoulos, E. Yu, "Using ontologies for knowledge management: an information system perspective", In Proceeding of 62nd Annual Meeting of the American Society for Information Science (ASIS99), pp. 482-496, 1999.

[69] A. Gomez Perez, M. Fernandez Lopez, O. Corcho, "Ontological Engineering", Springer – Verlag, London, 2004.

[70] M. Uschold, R. Jasper, "A Framework for Understanding and Classifying Ontology Applications", In Proceedings of IJCAI Workshop on Ontologies and Problems – Solving Methods, August 1999

[71] H. Chen, S. Fuller, W. Hersh, C. Friedman, "Medical informatics: Advances in knowledge management and data mining in biomedicine", Springer-Verlag; 2005.

[72] http://www.nlm.nih.gov/research/umls/

[73] http://mmtx.nlm.nih.gov/

[74] http://www.w3.org/TR/2004/REC-owl-guide-20040210/

[75] http://jena.sourceforge.net/

[76] http://clarkparsia.com/pellet/

[77] http://jena.sourceforge.net/ARQ/Tutorial/index.html

[78] http://www.ibm.com/developerworks/xml/library/j-sparql/

[79] http://www.w3.org/TeamSubmission/turtle/

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

[81] H. Wang, N. Noy, A. Rector, M. Musen, T. Redmond, D. Rubin, S. Tu, T. Tudorache, N. Drummond, M. Horridge, J. Seidenberg, "Frames and OWL Side by Side", 9th Intl. Protégé Conference, Stanford, California, July, 2006.

[82] http://www.mygrid.org.uk/OWL/Validator

[83] S. Sahay, B. Li, E. Garcia, E. Agichtein, and A. Ram, "Domain ontology construction from biomedical text", International conference on artificial intelligence, icai 2006.

[84] http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

[85] http://www.uwe-alex.de/Permutation/Permutation.java

[86] C. Calero, F. Ruiz, M. Piattini, "Ontologies for Software Engineering and Software Technology", Springer–Verlag, Berlin Heidelberg, 1st Edition, 2006.

ΣΥΝΤΟΜΟ ΒΙΟΓΡΑΦΙΚΟ

Ο Γεώργιος Λίτσιος γεννήθηκε στην Κοζάνη το 1983. Το 2000 ξεκίνησε τις σπουδές του στο Τμήμα Πληροφορικής του Πανεπιστημίου Ιωαννίνων,τις οποίες και ολοκλήρωσε τον Σεπτέμβριο του 2005. Το διάστημα Οκτώβριος του 2005 εώς τον Φεβρουάριο του 2009 παρακολούθησε το μεταπτυχιακό πρόγραμμα του ίδιου τμήματος, ενώ το διάστημα Ιούνιος του 2007 εώς σήμερα εργάζεται στο Τμήμα Διαχείρησης Πληροφοριακού Συστήματος της Επιτροπή Ερευνών του Πανεπιστημιού Ιωαννίνων.