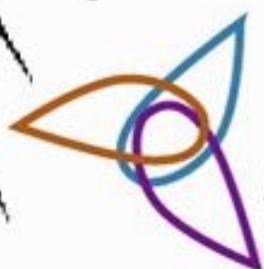
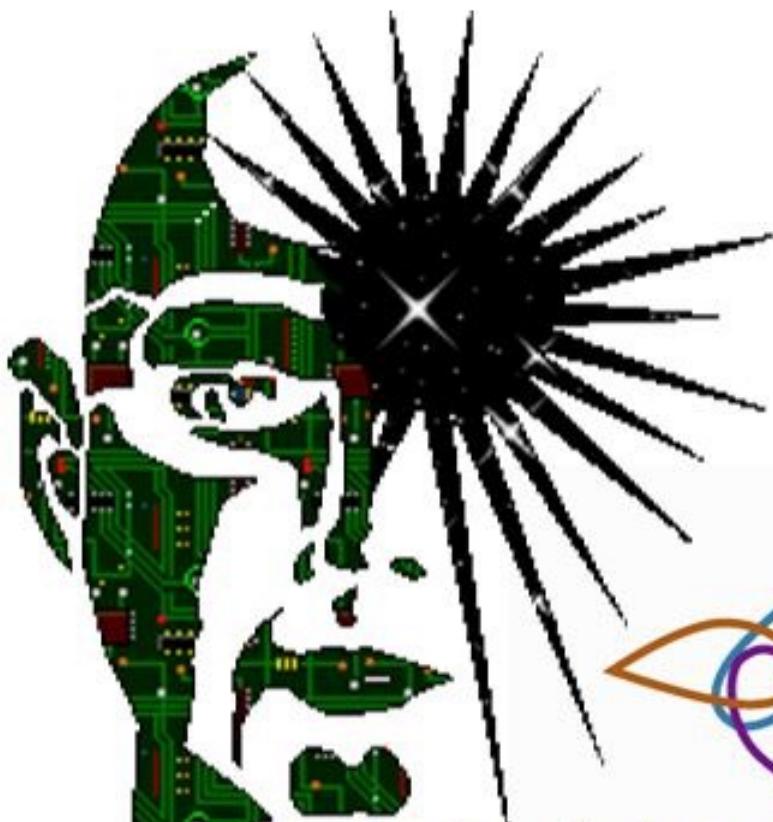


Συστήματα και Τεχνολογίες Γνώσης

2η Άσκηση



protégé

Συνεργάτες:

Ον/μο: Βαβουλιώτης Γεώργιος (ΑΜ: 03112083)

Ον/μο: Αθανασίου Νικόλαος (ΑΜ: 03112074)

Εξάμηνο: 8

Ημερομηνία Παράδοσης: 05/06/2016

Σκοπός της Άσκησης - Εισαγωγή : Στο πλαίσιο της άσκησης αυτής καλούμαστε να κατασκευάσουμε την σημασιολογική υποδομή για μια υπηρεσία που θα έχει ως στόχο να εξυπηρετεί ερωτήματα χρηστών σχετικά με κινηματογραφικές ταινίες. Συγκεκριμένα μας δίνεται μια οντολογία Ο για το πεδίο του κινηματογράφου μαζί με ένα ABox A , τα οποία έχουν προκύψει απευθείας από ένα μικρό απόσπασμα δεδομένων που δημοσιεύονται στον ιστότοπο της IMDb. Η συγκεκριμένη οντολογία και το ABox έχουν κατασκευαστεί υιοθετώντας άμεσα την μοντελοποίηση της υποκείμενης σχεσιακής βάσης και ως εκ τούτου δεν αποτελούν απαραίτητα «ορθή» εννοιολογική αναπαράσταση του συγκεκριμένου πεδίου γνώσης με βάση τις αρχές και τη σημασιολογία των περιγραφικών λογικών. Συγκεκριμένα η οντολογία περιλαμβάνει ενα σύνολο CN εννοιών(πχ Movie, Man), ενα σύνολο RN ρόλων(πχ hasActor, hasDirector, hasWriter) και ενα σύνολο ρόλων τύπου δεδομένων DN(title, runtime, name). Εμείς θα πρέπει να κατασκευάσουμε μια οντολογία η οποία να περιγράφει επαρκώς δεδομένα και να προσφέρει στους χρήστες την κατάλληλη ορολογία ώστε να μπορούν να υποβάλλουν τα ερωτήματα τους. Για τον σκοπό αυτό θα κάνουμε χρήση του Protégé, έτσι ώστε να τροποποιήσουμε έννοιες, ρόλους και τύπους δεδομένων στην αρχική βάση Ο, για να πετύχουμε το ζητούμενο.

Ερώτημα 1 : Περιγραφή των δυσκολιών που παρουσιάζει η οντολογία Ο, όσον αφορά τη χρήση της στη συγκεκριμένη εφαρμογή, χρησιμοποιώντας και συγκεκριμένα παραδείγματα.

Απάντηση: Αρχικά θα πρέπει να επισημάνω ότι η δοσμένη οντολογία Ο μαζί με το ABox δεν είναι σωστή εννοιολογικά, γεγονός το οποίο είναι προφανές. Από τα δεδομένα της άσκησης παρατηρώ ότι υπάρχουν κάποιες έννοιες και ρόλοι οι οποίοι είναι κάπως προβληματικοί, παρά το γεγονός ότι μας δίνετε η δυνατότητα να κάνουμε κάποιες αναζητήσεις που να έχουν νόημα. Η άσκηση αυτή ασχολείται με άτομα και ταινίες και είναι ορθότερο ο διαχωρισμός να γίνει αρχικά με βάση αυτά τα δυο και ύστερα να ορίσουμε διάφορες υποέννοιες για να αναπαραστήσουμε κάποια χαρακτηριστικά. Συγκεκριμένα λοιπόν, οι έννοιες και οι ρόλοι που παρουσιάζουν προβλήματα για την εφαρμογή αυτή είναι οι παρακάτω:

- **Country:** Η έννοια Country δεν είναι εννοιολογικά σωστή για την δική μας οντολογία διότι δεν μας ενδιαφέρει να έχουμε τις ίδιες τις χώρες σαν αντικείμενα του κόσμου μας αλλά θα ήταν καλύτερο να είχαμε τι ταινίες υπάρχουν με βάση τη χώρα. Για τον λόγο αυτό θα ήταν προτιμότερο για την οντολογία Ο' να ορίσουμε υποέννοιες του Movie με ονόματα AmericanMovie, OceanianMovie, EuropeanMovie και AsianMovie. Για την οντολογία Ο'' μπορούμε να αφαιρέσουμε τελείως την έννοια της χώρας και να κρατήσουμε την πληροφορία για τις ταινίες και τις υπόνοιες τους(το hasCountry μας είναι πλέον άχρηστο).
- **ColorType:** Η έννοια ColorType όπως μας δίνετε μας αναγκάζει να για κάθε ταινία πρέπει μέσω της σχέσης hasColor να βλέπουμε αν είναι έγχρωμη ή οχι. Μια καλύτερη ιδέα θα ήταν αντί να υλοποιούμε την παραπάνω ιδέα να δημιουργήσουμε δυο υπο-έννοιες τις έννοιας Movie με ονόματα BlackAndWhite και MovieColor και πλέον θα έχουμε την πληροφορία που χρειαζόμαστε για το χρώμα της κάθε ταινίας.

Προφανώς μετά από την προσθήκη αυτή η έννοια ColorType και ο ρόλος hasColorType δεν είναι πλεον χρήσιμα και μπορούμε να τα αφαιρέσουμε.

- **Language:** Και για την έννοια Language θα μπορούσαμε να ισχυριστούμε οτι είναι άχρηστη αφού μας δίνει πληροφορία την οποία θα μπορούσαμε να την πάρουμε ορίζοντας κάποιες υπο-έννοιες του Movie οι οποίες θα αναφέρονται στις γλώσσες στις οποίες χρησιμοποιούνται στην εκάστοτε ταινία. Συγκεκριμένα τέτοιες υπο-έννοιες είναι οι EnglishLanguage GreekLanguage κτλ. Αν λοιπόν προχωρήσουμε με μια τέτοια ενέργεια ο ρόλος hasLanguage θα μας είναι πλέον περιττός και θα μπορούσαμε να την διαγράψουμε, αλλα είναι περιττό να το κανουμε για τοσους πολλους όρους.
- **Genre:** Με την ίδια λογική με παραπάνω μπορούμε να προσθέσουμε διάφορες υποέννοιες στην έννοια Movie με βάση το Genre μιας ταινίας, αφού το Genre αφορά μόνο ταινίες. Χαρακτηριστικά παραδείγματα είναι είναι τα Comedy,Drama,Action κτλ. Προφανώς πλέον ο ρόλος hasGenre είναι άχρηστος και μπορούμε να τον διαγράψουμε όπως και οι έννοια Genre.
- **Year:** Η έννοια Year με ακριβώς την ίδια λογική θα μπορούσε να διαγραφεί με την προσθήκη υπο-έννοιών της έννοιας Movie αφού πλέον θα είναι άχρηστο. Χαρακτηριστικά παραδείγματα είναι είναι τα 2001Movie, 2002Movie, 2003Movie κλπ. Προφανώς πλέον ο ρόλος hasProductionYear είναι άχρηστος και μπορούμε να τον διαγράψουμε αν το κανουμε για ολες τις χρονιες, αλλα είναι περιττό.
- **Σημαντικότερα Εννοιολογικά Λάθη:** Τέλος κατά την γνώμη μας τα σημαντικότερα εννοιολογικά λάθη είναι αρχικά η παρουσία των εννοιών Man και Woman σαν υποκλάσεις του Thing και δεύτερον η καθολική έλλειψη εννοιών Actor, Director και Writer. Αρχικά ας εξετάσουμε το πρώτο εννοιολογικό λάθος το οποίο θα μπορούσε να διορθωθεί με την προσθήκη μιας έννοιας με όνομα Person σαν υποκλάση του Thing η οποία είναι μια πιο γενική έννοια για να εκφράσουμε την έννοια του ατόμου. Το δεύτερο εννοιολογικό λάθος είναι πολύ γενικό διότι τα προβλήματα που δημιουργεί είναι αρκετά. Αρχικά οι έννοιες Actor, Director και Writer δεν ορίζονται καθόλου αλλό μόνο οι ρόλοι has Actor, hasDirector και hasWriter οι οποίοι στην ουσία αυτό που κάνουν είναι να συνδέουν ενα άτομο με μια ταινία αλλά χωρίς ανιχνεύουν την αντίστροφη σχέση πράγμα το οποίο θα θέλαμε. Για το λόγο αυτό θα προσθέσουμε τις έννοιες Actor, Director και Writer ώστε κάθε άτομο να μπορεί να έχει και αυτές τις ιδιότητες καθώς και τις αντίστροφες σχέσεις από αυτές που υπάρχουν ώστε η αναζήτηση να είναι σωστή και πλήρης.

Ερώτημα 2 : Η νέα οντολογία Ο' (σε μορφή Περιγραφικών Λογικών και .owl) την οντολογία Ο', καθώς και μία περιγραφή της, σε σχέση και με τα προβλήματα που παρουσιάζει η οντολογία Ο.

Στο ερώτημα αυτό ουσιαστικά θα εφαρμόσουμε όλα όσα αναφέραμε θεωρητικά στο Ερώτημα1. Αφού κάνουμε τα απαιτούμενα, τα οποία θα περιγράψω αναλυτικά παρακάτω, θα έχουμε δημιουργήσει πλέον μια νέα οντολογία Ο' η οποία θα είναι εννοιολογικά σωστή και δεν θα έχει τα προβλήματα που αναφέραμε. Για την υλοποίηση αυτών χρησιμοποιήσαμε το εργαλείο Protege. Παρακάτω θα σας παρουσιάσω εκτενώς όλα όσα προσθέσαμε και αφαιρέσαμε (έννοεις και ρόλους) στην οντολογία Ο για να πάρουμε την οντολογία Ο' (στο προηγούμενο ερώτημα σας παρουσίασα αναλυτικά τις διάφορες αλλαγές που πρέπει να γίνουν και εδώ τις αναφέρω ενδεικτικά και δείχνω τον τρόπο υλοποίησης):

- Δημιουργούμε τις έννοιες AmericanMovie, OceanianMovie, EuropeanMovie και AsianMovie σαν υπο-έννοιες της έννοιας Movie αφού η έννοια Country δεν είναι εννοιολογικά σωστή για την δική μας οντολογία διότι δεν μας ενδιαφέρει να έχουμε τις ίδιες τις χώρες σαν αντικείμενα του κόσμου μας αλλά θα ήταν καλύτερο να είχαμε τις ταινίες υπάρχουν με βάση τη χώρα. Για να το κάνουμε αυτό κάνουμε τα εξής :

Για το EuropeanMovie :

subclass of Movie

hasCountry value France or hasCountry value Greece (για όλες τις Ευρωπαϊκές Χώρες)

Για το AmericanMovie :

subclass of Movie

hasCountry value USA or hasCountry value Brasil (για όλες τις Αμερικανικές Χώρες)

Ακολουθώ την ίδια διαδικασία για όλες τις υπόλοιπες ηπείρους αλλά δεν την γράφω αναλυτικά αφού είναι προφανές το αποτέλεσμα.

- Δημιουργούμε τις έννοιες BlackAndWhite και ColorMovie σαν υπο-έννοιες της έννοιας Movie αφού θα έχουμε άμεσα την πληροφορία για το χρώμα μιας εικόνας χωρίς την χρήση ρόλων. Για να το κάνουμε αυτό κάνουμε τα εξής :

Για το ColorMovie :

subclass of Movie

hasColorType value Color

Για το BlackAndWhite :

subclass of Movie

hasColorType value Black_and_White

Μέτα θα πρέπει να διαγράψουμε την έννοια Color και τον ρόλο hasColor αφού μας είναι άχρηστα.

- Δημιουργούμε τις έννοιες Comedy,Drama,Action κτλ σαν υπο-έννοιες της έννοιας Movie αφού το Genre αναφέρεται αποκλειστικά σε ταινίες. Για παραπάνω εξήγηση δες το ερώτημα 1. Για να το κάνουμε αυτό κάνουμε τα εξής :

Για την **Action** :

hasGenre value Action
subclass of Movie

Για την **Drama**:

hasGenre value Drama
subclass of Movie

Το ίδιο ακριβώς κάνουμε και για τις υπόλοιπες έννοιες comedy κλπ αλλά δεν τα παραθέτω όλα λόγω όγκου και ομοιότητας.

Μέτα θα πρέπει να διαγράψουμε την έννοια Genre και τον ρόλο hasGenre αφού μας είναι άχρηστα.

- Δημιουργούμε την έννοια Person, η οποία θα είναι υποκλάση του Thing για να είναι πιο ορθή η οντολογία μας και να μπορέσουμε να ορίσουμε στη συνέχεια τις έννοιες Director, Actor και Writer. Η Person θα έχει σαν υπο-έννοιες της τις έννοιες Man και Woman και ουσιαστικά θα ορίζει πιο σωστά και γενικά την έννοια του ατόμου. Η έννοια Person έχει τον εξής τύπο: Man or Woman
- Δημιουργούμε τις έννοιες Director, Actor και Writer για να μπορούμε να έχουμε για κάθε άνθρωπο-άτομο(Person) σε ποιες κατηγορίες ανήκει και σε ποιές ταινίες έιναι για παράδειγμα σκηνοθέτης, ηθοποιός, συγγραφέας. Αυτό το πετυχαίνουμε με την χρήση των ήδη υπάρχοντων ρόλων hasDirector, hasWriter και hasActor μαζί με την βοήθεια 3 νέων ρόλων που ορίζω με ονόματα isDirector , isWriter , isActor οι οποίοι είναι οι αντίστροφοι ρόλοι από τους hasDirector, hasWriter και hasActor αντίστοιχα. Για τη δημιουργία τους κάνουμε τα εξής :

Για τον **Director**:

isDirector some Movie
subclass of Person

Για τον **Actor**:

isActor some Movie
subclass of Person

Για τον Writer:

isWriter some Movie

subclass of Person

Στη συνέχεια σας παραθέτω το Tbox της οντολογίας Ο' πριν να διαγράψουμε τις άχρηστες πλεον έννοιες και ρόλους :

Tbox = {

Person ≡ Man ∪ Woman

Man ⊑ Person

Woman ⊑ Person

Actor ⊑ Person

Director ⊑ Person

Writer ⊑ Person

Man ≡ Woman

isDirector ≡ hasDirector

isWriter ≡ hasWriter

isActor ≡ hasActor

Director ≡ \exists hasDirector.Movie ∩ Person

Actor ≡ \exists hasActor.Movie ∩ Person

Writer ≡ \exists hasWriter.Movie ∩ Person

Action ≡ \exists hasGenre.Action ∩ Movie

Drama ≡ \exists hasGenre.Drama ∩ Movie

Comedy ≡ \exists hasGenre.Comedy ∩ Movie

Adventure ≡ \exists hasGenre.Adventure ∩ Movie

(δεν γράφω για όλα τα Genres => όμοια για τα υπόλοιπα)

ColorMovie ≡ \exists hasColor.Color ∩ Movie

BlackAndWhiteMovie ≡ \exists hasColor.Black_and_White ∩ Movie

EuropeanMovie ≡ (\exists hasCountry.Greece ∪ \exists hasCountry.Italy ∪ ...) ∩ Movie

AsianMovie ≡ (\exists hasCountry.Japan ∪ \exists hasCountry.China ∪) ∩ Movie

AmericanMovie ≡ (\exists hasCountry.USA ∪ \exists hasCountry.Brasil ∪) ∩ Movie

OceanianMovie ≡ (\exists hasCountry.Australia ∪ \exists hasCountry.Tanzania ∪) ∩ Movie

}

Ερώτημα 3 : Η νέα οντολογία Ο”(σε μορφή Περιγραφικών Λογικών και .owl) την οντολογία Ο’, καθώς και μία περιγραφή της, σε σχέση και με τα προβλήματα που παρουσιάζει η οντολογία Ο.

Στο τρίτο μέρος της άσκησης καλούμαστε να πάρουμε την οντολογία του Ερωτηματος 2, δηλαδή την Ο’ και να τις προσθέσουμε διάφορες νέες έννοιες ώστε να κάνουμε την αναζήτηση πιο πλούσια από άποψη δυνατοτήτων. Στη συνέχεια σας παρουσιάζω τις έννοιες τις οποίες πρόσθεσα και την αναλυτική τους περιγραφή:

- **2001Movie, 2002Movies, 2003Movies κ.ο.κ:** Στην οντολογία Ο’ έχουμε μια έννοια Year και ένα ρόλο hasProductionYear. Αντ’ αυτού, θα μπορούσαμε να δημιουργούμε υποέννοιες του Movie με βάση το Year που έχουν. Εμεις το κανουμε για ορισμένες χρονιές και πολύ εύκολα επεκτείνεται ώστε να παραλειφθούν οι παραπάνω όροι. Έτσι δημιουργήσαμε τις υποέννοιες με βάση τη παρακάτω λογική: Για το διαχωρισμό με βάση το Year:

$2001_Movie \equiv \text{hasProductionYear value year_2001 and Movie (κ.ο.κ)}$

- **EnglishLanguage GreekLanguage κτλ :** Στην οντολογία Ο” θα μπορούσαμε να εχουμε διαγράψει την έννοια Language και το ρόλο hasLanguage και στη θέση τους έχουμε φτιάξει υποέννοιες του Movie με βάση τη γλώσσα που μιλάνε στη ταινία αν το καναμε για ολες τις γλωσσες. Εμεις το καναμε για τις βασικές γλώσσες που θεωρουμε οτι θα αναζητησει ο μεσος χρήστης, αλλα πολύ ευκολα επεκτεινεται και για ολες. Πιο συγκεκριμένα ακολουθήσαμε τη παρακάτω λογική:

- $\text{GreekLanguage} \equiv \text{Movie and hasLanguage value Greek (κ.ο.κ)}$

- **ActorInComedy, DirectorInSci-Fi:** Ορίζοντας τις παραπάνω έννοιες μπορούμε να επιτρέψουμε στο χρήστη να ψάξει για καλλιτέχνες που ασχολούνται με συγκεκριμένα είδη ταινιών, έτσι ώστε να μπορεί να βρεί ευκολότερα κάποιο καλλιτέχνη των προτιμήσεων του. Οι έννοιες αυτές υλοποιήθηκαν με τον εξής τρόπο:

- $\text{ActorInComedy} \equiv \text{Actor and isActor some Comedy}$
- $\text{DirectorInSci-Fi} \equiv \text{Director and isDirector some Sci-Fi}$

Για την οντολογία Ο” **πρόσθεσα στο Tbox τις παρακάτω έννοιες :**

TBox = {

2001Movie = Movie \sqcap \exists hasProductionYear.year_2001

κ.ο.κ.

2001Movie \sqsubseteq Movie (αναλόγως και για τις άλλες χρονιές)

GreekLanguage= Movie \sqcap \exists hasLanguage.Greek

(αναλόγως έγιναν και οι υπόλοιπες έννοιες που αφορούν το Language)

ActorInComedy = Actor \sqcap \exists isActor.Comedy

DirectorInSci-Fi = Director \sqcap \exists isDirector.Sci-Fi

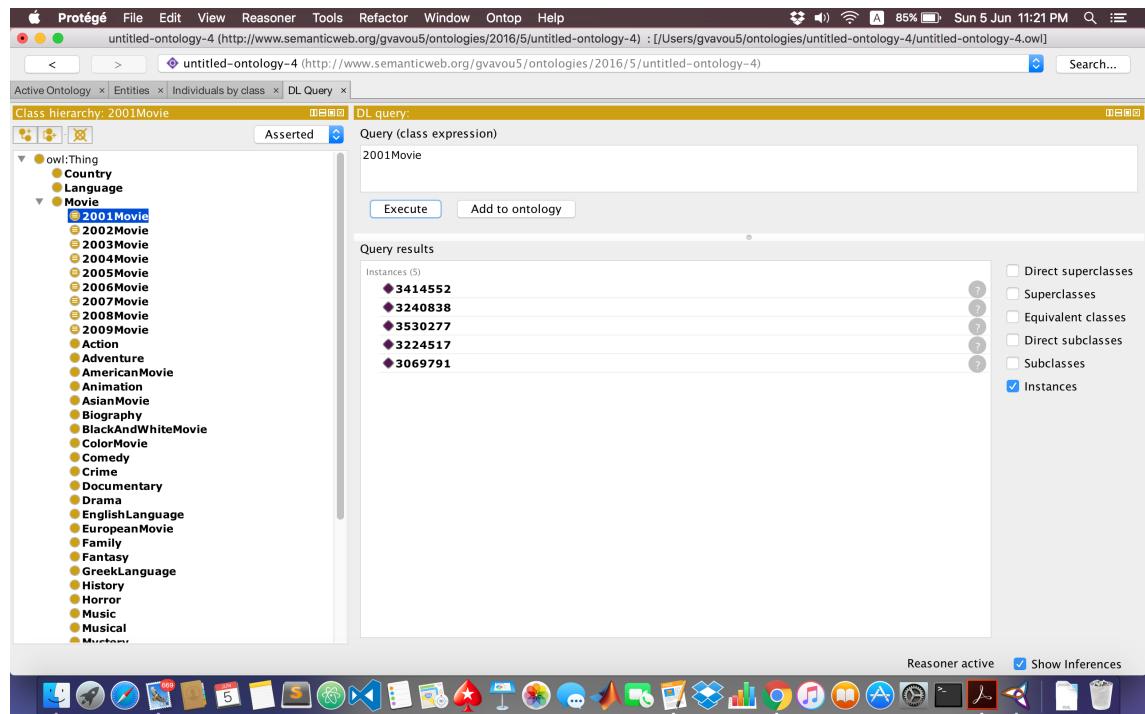
}

Σχολιασμός : Πλέον η οντολογία έχει μέσα περισσότερες έννοεις και η αναζήτηση για τον κάθε χρήστη είναι καλύτερη. Θα πρέπει να παρατηρήσω ότι θα μπορούσα να εισάγω όλες τις πιθανές γλώσσες ώστε να διώξω την σχέση hasLanguage αλλά επέλεξα να μην το κάνω αλλά να το αναφέρω απλά.

Ερώτημα 4 : Ένα σύνολο πιθανών ερωτημάτων χρήστη, που θα βασίζονται σε πιθανά σενάρια χρήσης, καθώς και τις απαντήσεις τους (να σημειωθούν οι επιπλέον απαντήσεις που παίρνουμε από τη χρήση υπηρεσιών συλλογιστικής για την απάντηση των ερωτημάτων).

Πιθανά Ερωτήματα Χρήστη :

- 1) Το παρακάτω query επιστρέφει τις ταινίες που έχουν έτος παραγωγής το 2001 :



2) Το παρακάτω query επιστρέφει τις ταινίες που έχουν έτος παραγωγής το 2002 και είναι κωμωδίες :

The screenshot shows the Protégé interface with the following details:

- Class hierarchy:** 2001Movie
- DL query:** Comedy **and** 2002Movie
- Query results:** Instances (3)
 - 3096702
 - 2875436
 - 2715537
- Filter checkboxes:** Direct superclasses, Superclasses, Equivalent classes, Direct subclasses, Subclasses, **Instances** (selected)
- Buttons:** Execute, Add to ontology
- Toolbar:** Includes icons for file operations, search, and reasoner status.

3) Το παρακάτω query επιστρέφει όλους τους άνδρες ηθοποιούς που παίζουν σε κωμωδία:

The screenshot shows the Protégé interface with the following details:

- Class hierarchy:** Person
- DL query:** ActorInComedy **and** Man
- Query results:** Instances (235)
 - 1251509
 - 1296414
 - 1871716
 - 1791956
 - 233402
 - 1706457
 - 414348
 - 2320025
 - 93508
 - 2232636
 - 1855224
 - 861240
 - 1439083
 - 201807
 - 1337053
 - 1068721
 - 1756705
 - 973077
 - 381323
- Filter checkboxes:** Direct superclasses, Superclasses, Equivalent classes, Direct subclasses, Subclasses, **Instances** (selected)
- Buttons:** Execute, Add to ontology
- Toolbar:** Includes icons for file operations, search, and reasoner status.

4) Το παρακάτω query επιστρέφει όλες τις ταινίες που είναι ασπρόμαυρες και η γλώσσα τους είναι η Αγγλική :

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, Ontop, Help, and a status bar indicating Sun 5 Jun 11:22 PM. The title bar says "untitled-ontology-4 (http://www.semanticweb.org/gvavou5/ontologies/2016/5/untitled-ontology-4) : [/Users/gvavou5/ontologies/untitled-ontology-4/untitled-ontology-4.owl]". The main window has tabs for Active Ontology, Entities, Individuals by class, and DL Query. The DL Query tab is active, showing a query expression: "BlackAndWhiteMovie and EnglishLanguage". Below the query is an "Execute" button and an "Add to ontology" button. The results pane displays "Query results" with "Instances (7)" listed: 2689274, 3512641, 3006212, 3394393, 3091665, 2547910, and 3317242. To the right of the results are checkboxes for filtering: Direct superclasses, Superclasses, Equivalent classes, Direct subclasses, Subclasses, and Instances, with Instances checked. The bottom of the screen shows the Mac OS X dock with various application icons.

5) Το παρακάτω query επιστρέφει την ταινία με τίτλο ο “The Mexican” :

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, Ontop, Help, and a status bar indicating Sun 5 Jun 11:23 PM. The title bar says "untitled-ontology-4 (http://www.semanticweb.org/gvavou5/ontologies/2016/5/untitled-ontology-4) : [/Users/gvavou5/ontologies/untitled-ontology-4/untitled-ontology-4.owl]". The main window has tabs for Active Ontology, Entities, Individuals by class, and DL Query. The DL Query tab is active, showing a query expression: "title value \"The Mexican\"". Below the query is an "Execute" button and an "Add to ontology" button. The results pane displays "Query results" with "Instances (1)" listed: 3530277. To the right of the results are checkboxes for filtering: Direct superclasses, Superclasses, Equivalent classes, Direct subclasses, Subclasses, and Instances, with Instances checked. The bottom of the screen shows the Mac OS X dock with various application icons.

Protégé File Edit View Reasoner Tools Refactor Window Ontop Help

untitled-ontology-4 (<http://www.semanticweb.org/gvavou5/ontologies/2016/5/untitled-ontology-4>) : [/Users/gvavou5/Desktop/scr2.owl]

Active Ontology Entities Individuals by class DL Query

Class hierarchy: Movie Individual Annotations Individual Usage

Annotations: 3530277

Annotations +

Description: 3530277 Property assertions: 3530277

Types + Object property assertions +

2001Movie hasActor 3223502
 Adventure hasActor 3394156
 AmericanMovie hasLanguage English
 ColorMovie hasLanguage Spanish
 Comedy hasActor 134689
 Crime hasActor 115935
 EnglishLanguage hasActor 615537
 Movie hasActor 1298435
 Romance hasActor 18823
 hasActor 2715003
 hasActor 95576
 hasActor 1660143
 hasActor 397995
 hasActor 1272354
 hasActor 4E7171

Assered in: <http://www.semanticweb.org/gvavou5/ontologies/2016/5/untitled-ontology-4>

Individuals by type Annotation property hierarchy Datatypes

Object property hierarchy Data property hierarchy

Object property hierarchy: asserted

owl:topObjectProperty

Reasoner active Show Inferences

6) Το παρακάτω query επιστρέφει όλες τις γυναίκες σκηνοθέτες σε Sci-Fi ταινίες :

Protégé File Edit View Reasoner Tools Refactor Window Ontop Help

untitled-ontology-4 (<http://www.semanticweb.org/gvavou5/ontologies/2016/5/untitled-ontology-4>) : [/Users/gvavou5/ontologies/untitled-ontology-4/untitled-ontology-4.owl]

Active Ontology Entities Individuals by class DL Query

Class hierarchy: Writer DL query:

Query (Class expression): DirectorInSci-Fi and Woman

Execute Add to ontology

Query results

Instances (2)

- 2894375
- 3620907

Reasoner active Show Inferences

7) Το παρακάτω query επιστρέφει όλες τις έγχωρες ταινίες οι οποίες έιναι ταυτόχρονα και Crime και Mystery :

The screenshot shows the Protégé 4.0 interface with the following details:

- Class hierarchy:** Country
- DL query:** Crime and Mystery and ColorMovie
- Query results:** Instances (6)
 - 2689274
 - 3347760
 - 3360980
 - 2731260
 - 3445495
 - 3472683
- Filter checkboxes:** Direct superclasses, Superclasses, Equivalent classes, Direct subclasses, Subclasses, Instances (checked)
- Toolbar:** Includes icons for various applications like Finder, Mail, Safari, and Xcode.

8) Το παρακάτω query επιστρέφει όλες τις ευρωπαϊκές ταινίες που έχουν ελληνική γλώσσα :

The screenshot shows the Protégé 4.0 interface with the following details:

- Class hierarchy:** Writer
- DL query:** GreekLanguage and EuropeanMovie
- Query results:** Instances (1)
 - 2613514
- Filter checkboxes:** Direct superclasses, Superclasses, Equivalent classes, Direct subclasses, Subclasses, Instances (checked)
- Toolbar:** Includes icons for various applications like Finder, Mail, Safari, and Xcode.

Ερώτημα 5 :Μία ανάλυση των περιορισμών που συναντήσατε κατά την περιγραφή του παραπάνω πεδίου γνώσης οι οποίοι απορρέουν από την εκφραστικότητα των περιγραφικών λογικών (περίπου 1 σελίδα).

Περιγραφή Προβλημάτων-Περιορισμών: Σε αυτό το κομμάτι της άσκησης, θα περιγράψουμε τα βασικότερα προβλήματα που αντιμετωπίσαμε καθώς και πως προσπαθήσαμε να τα ξεπεράσουμε. Τα βασικά προβλήματα που θα αναφέρουμε απορρέουν από την εκφραστικότητα και ιδιαιτερότητα των περιγραφικών λογικών. Τα περισσότερα από αυτά προέρχονται από το λεγόμενο Open World Assumption. Αυτό πρακτικά σημαίνει πως μόνο όταν έχουμε κάποια πληροφορία για ένα άτομο, μπορούμε να τη θεωρήσουμε έγκυρη/σίγουρη αν είναι στην οντολογία. Σε αντίθεση με το closed world assumption μονού όταν μια πληροφορία αναφέρεται και ορίζεται ρητά για κάποιο άτομο στην οντολογία τότε μπορούμε να συμπεράνουμε ότι αυτό έχει καποια ιδιότητα αλλιώς δεν μας δίνει όχι αλλά θεωρείται η πληροφορία ως ανύπαρκτη. Πρέπει να δηλώνουμε ρητά όσα άτομα δεν έχουν ή έχουν μια ιδιότητα καθώς μια οντολογία περιγραφικής λογικής(και ο reasoner που χρησιμοποιούμε) αν δεν μια τέτοια δήλωση δεν θεωρεί ότι δεν έχουν την ιδιότητα θεωρεί ότι μπορεί να συμβαίνει το οτιδήποτε δηλαδή είτε να την έχουν είτε όχι την στιγμή που ένα query πχ SQL για την βάση χρησιμοποιεί closed world assumption και αν δεν βρει κάποιο άτομο να την έχει επιστρέφει όχι. Στη συνέχεια παρατίθενται κάποιοι όροι του περιβάλλοντος Protege που μας δυσκόλεψαν και αποδείχθηκε εις βάρος μας η ελαστική εκφραστικότητα που προσφέρει η περιγραφική λογική.

Αρχικά η χρήση του ορού max από τη στιγμή που δεν οριζόταν ρητά κάποιο πάνω όριο δεν επέστρεφε απάντηση αφού θεωρούσε ότι μπορεί να υπάρχουν και άλλα άτομα που να πληρούν την ιδιότητα max πάνω στη σχέση πχ R τα οποία δεν έχουν οριστεί επομένως δεν αρκούσε να τρέξουμε max αλλά να ορίσουμε και καποιο πάνω όριο ώστε να μας δώσει αποτέλεσμα. Στη συνέχεια το exactly x επέβαλε με παρόμοιο τρόπο με το max την εύρεση ακριβώς χ φορές του όρου χ και χρειάστηκε ορισμός κατάλληλου εύρους ώστε να υπεβληθεί. Η χρήση του only x γυρνούσε άτομα με παρονσία της ιδιότητας μόνο μίαφορά και καμία άλλη το οποίο ήταν σχετικά δύσχρηστο στην περίπτωση του είδους της ταινίας καθώς η ιδιότητα είδος από μονη της μπορούσε να υπακούει σε περισσότερες από μία τιμές γεγονός που το έκανε να αποτυγχάνει κατά την αναζήτηση παρότι υπήρχαν ταινίες με τέτοια ιδιότητα ο ορισμός του δεν επεβαλε ρητά την ύπαρξη ενός είδους. Η χρήση του not. Η αδυναμία χρήσης της άρνησης(not) (λόγο του open world assumption) μας δημιούργησε πολλά προβλήματα. Δεν μπορούσαμε να ορίσουμε ‘αντίθετους’ ρόλους, αλλά υπήρχαν πολλά πράγματα στα οποία μας περιόρισε το not αλλά ανέφερα παραπάνω τα πιο σημαντικά διότι στα άλλα ζητήματα βρήκαμε λύση.

Θετικά του open world:

- Underspecification δηλαδή αφημερημένες εμφωλευμένες και ανώνυμες οντότητες.
- Εύκολα επαναχρησιμοποιήσιμος και επεκτάσιμος όπως πράξαμε κατά την κατασκευή των οντολογιών O' και O'' από την O και την κατανόηση των προβλημάτων της.
- Καλός στην επεξεργασία γνώσης λόγω ας πούμε της χρήσης του equivalent.
- Βοηθά κατά την επεξεργασία βάσεων και γνώσεν με καλή ανταπόκριση στην πραγματικότητα και σωστό αντικατοπτρισμό της αφού ας πούμε στο πεδίο των επιστημών όταν μια πληροφορία λείπει είναι απλώς άγνωστη όχι δεν υπάρχει.

Αρνητικά open world :

- Καποια queries και προβλήματα είναι από τη φύση τους καθαρά κλειστού κόσμου.
- Δεν βοηθάει στον έλεγχο εγκυρότητας των δεδομένων και στην επιβολή περιορισμών.
- Δεν επιτρέπει σωστό χειρισμό εξαιρέσεων.

Παράδειγμα από Wikipedia

Example(wikipedia)

Statement: "Mary" "is a citizen of" "France"

Question: Is Paul a citizen of France?

"Closed world" (for example Prolog) answer: No.

"Open world" answer: Unknown.