Μοντελοποίηση Πληροφοριακού Συστήματος Γυμνασίου/Λυκείου

Ηρακλής Θεοφανίδης
Τμήμα Πληροφορικής Και Ηλεκτρονικών Συστημάτων
Θεσσαλονίκη, Κεντρική Μακεδονία, Ελλάδα
iraklistheofanidis@gmail.com

Αχιλλέας Γκέκας
Τμήμα Πληροφορικής Και Ηλεκτρονικών Συστημάτων
Θεσσαλονίκη, Κεντρική Μακεδονία, Ελλάδα
gkekas57@gmail.com

Ερρίκος Καλτσόπουλος
Τμήμα Πληροφορικής Και Ηλεκτρονικών Συστημάτων
Θεσσαλονίκη, Κεντρική Μακεδονία, Ελλάδα
ekal638@gmail.com

Περίληψη
Στο παρόν έγγραφο περιγράφουμε αναλυτικά τον τρόπο μοντελοποίησης λυκείου/γυμνασίου μέσω του Σημασιολογικού Ιστού. Ξεκινώντας αναλύουμε κάποιες βασικές κλάσεις και τις συσχετίσεις από τις οποίες αποτελείται το οργανόγραμμα ενός σχολείου. Τα εργαλεια τα οποία χρησιμοποιήθηκαν για την συγκεκριμένη μοντελοποίηση είναι το Protege και η SparQL.

Protege
Το Protege είναι ένα δωρεάν, Open Source πρόγραμμα που βοηθάει στην σύνταξη οντολογιών του σημασιολογικού Ιστού. Υποστηρίζεται ενεργά από μια ισχυρή κοινότητα χρηστών, προγραμματιστών, ακαδημαϊκών, κυβερνητικών και εταιρικών χρηστών που θέτουν ερωτήσεις, γράφουν τεκμηρίωση και συνεισφέρουν plug-ins με στόχο τη δημιουργία λύσεων βασισμένων στη γνώση σε διαφορετικές περιοχές όπως η βιοϊατρική, το ηλεκτρονικό εμπόριο. Το Protégé υποστηρίζει πλήρως τις τελευταίες προδιαγραφές OWL 2 Ontology Language και RDF. Βασίζεται στην Java, είναι επεκτάσιμο και παρέχει ένα περιβάλλον plug-and-play που το καθιστά μια ευέλικτη βάση για ταχεία ανάπτυξη πρωτοτύπων και εφαρμογών.

Sparql

Η sparql ειναι μια γλωσσα ερωτησεων η οποία επιτρέπει την ανάκτηση και τον χειρισμό δεδομένων που είναι αποθηκευμένα στη μορφή του RDF.

Keywords

Protege,Sparql,School,Semantic web.

Classes
Επειδή το θέμα μας είναι το οργανόγραμμα ενός σχολειου με το βασικό συστατικό που το απαρτίζει να είναι οι άνθρωποι , δημιουργήσαμε μια κλάση People. Ένα σχολείο όμως αποτελείται και από άλλες βασικές κλάσεις όπως: Τάξη (taksi), Μαθήματα (mathimata), Βαθμολογία (vathmologia), Εκδρομές (ekdromes). Έτσι πιο αναλυτικά προέκυψαν οι εξής κλάσεις-υποκλάσεις:

- **People**
  - **Student**(μαθητές)
  - **Teachers**(καθηγητές)
  - **Prosopiko**(προσωπικό)Στο προσωπικό ειναι φυλακας ,υπευθυνος κυλικειου.
  - **SullogosGonewn**(είναι ο σύλλογος γονέων και κηδεμόνων οι οποίοι παίρνουν αποφάσεις για θέματα του σχολείου)
- **Taksi**(κάθε μαθητής ανήκει σε μια από της τάξεις)
  - **Prwth**
  - **Deutera**
  - **Trith**
- **Mathimata**(κάθε σχολείο αποτελείται από κάποια μαθήματα που παρακολουθούν οι μαθητές)
- **Vathmologia**(κάθε μαθητής βαθμολογείται ανάλογα με τις επιδόσεις του στα μαθήματα)
- **Ekdromes**
  - **Anapsixis**
  - **Ekpaideutikh**
- **Gender**
  - **Male**(Άνδρας)
  - **Female**(Γυναίκα)

Properties
Το Protege μας δίνει την δυνατότητα να δώσουμε κάποιες ιδιότητες στα αντικείμενα μας οι οποίες χωρίζονται σε δύο κατηγορίες Object Properties και Data Properties. Οι δύο αυτές ιδιότητες μοιάζουν πολύ κατά την δημιουργία τους αλλά διαφέρουν στην λειτουργικότητα τους.Η πιο βασική διαφορά είναι αυτή που δίνουμε στα range. Επίσης το object property μας δίνει κάποιες επιπλέον επιλογές.

Data Properties
- **hasBirthYear**:Το κάθε Individual που δημιουργείται ως People έχει μια ημερομηνία γέννησης.

- **hasCourseName**:Το κάθε μάθημα έχει κάποιο όνομα.
- **hasDescription**:Η κάθε εκδρομή του σχολείου αποτελείται από μια περιγραφή-πρόγραμμα για την ενημέρωση γονέων και μαθητών.
- **hasDate**:Είναι η ημερομηνία διεξαγωγής των εκδρομών.
- **hasLocation**:Είναι η τοποθεσία που θα πάνε εκδρομή.
- **hasName**:Κάθε άνθρωπος έχει όνομα.
- **hasSurname**:Κάθε άνθρωπος έχει επώνυμο.
- **hasWrario**:Οι μαθητές και γενικά όσοι δουλεύουν σε κάποιο σχολείο έχουν κάποιο ωράριο.
- **isAlso**:Κάποιο άτομο μπορεί να έχει και κάποια άλλη ιδιότητα (π.χ. ένας καθηγητής μπορεί να είναι και Διευθυντής).
- **isMemberOf**: Ένας μαθητής μπορεί να είναι μέλος κάποιας ομάδας του σχολείου ή του 15μελούς.
- **isA**: Δηλώνει την ιδιότητα κάθε μέλους από το προσωπικό.
- **hasTmhma** βάζουμε το όνομα του τμήματος σε String μορφή στο individual
- **hasGradeSTR** Περνάμε τον βαθμό σε String μορφή στο individual

Object Properties

- **hasClass** (Domain Student ranges Taksi και reverse of hasStudent) Ένας μαθητής ανήκει σε μια τάξη.
- **hasChild** (Domain SullogosGonewn ranges Student Και inverse of hasParent) Ο κάθε γονέας που επιθυμεί να ανήκει στον σύλλογο γονέων πρέπει να έχει κάποιο παιδί που ανήκει στο σχολείο.
- **hasCourses** (Domain Student ranges Mathimata) Ένας μαθητής έχει κάποια μαθήματα.
- **hasGrade** (Domain Student ranges Vathmologia) Η επίδοση ενός μαθητή φαίνεται μέσω της βαθμολογίας του.
- **hasParent** (Domain Student ranges Sullogos Gonewn και inverse του hasChild) Ο κάθε μαθητής έχει κάποιο γονιό.
- **hasStudent** (Domain Taksi ranges Student και inverse of hasClass) Η κάθε τάξη αποτελείται από κάποιους μαθητές.
- **hasTrips** (Domain Taksi ranges Ekdromes) Η κάθε τάξη πάει κάποιες εκδρομές.
- **isOrganazing** (Domain Teachers ranges Ekdromes και inverse of isOrganized) Οι καθηγητές οργανώνουν κάποια εκδρομή.
- **hasResponsibleTeacher** (Domain Taksi ranges Teachers Και Inverse of isResponsible) Η κάθε τάξη έχει υπεύθυνο καθηγητή.
- **isOrganized** (Domain Ekdromes ranges Teachers και Inverse of IsOrgranazing) Οι εκδρομές οργανώνονται από κάποιους καθηγητές.
- **isResponsible** (Domain ranges και inverse Of hasResponsibleTeacher) Ένας καθηγητής μπορεί να είναι υπεύθυνος για μια τάξη.
- isTeaching (Domain Teachers ranges Mathimata και inverse of isTeachedBy) Ένας καθηγητής διδάσκει ένα μάθημα.
- isTeachedBy (Domain Mathimata ranges Teachers και inverse of isTeaching) Ένα μάθημα διδάσκεται από κάποιον καθηγητή.

Axioms

Τα αξιώματα (axioms) τα χρησιμοποιούμε για να δώσουμε κάποιους περιορισμούς στις κλάσεις μας:

- **Ekdromes**
  - **isOrganized max 3 Teachers** (μια εκδρομή οργανώνεται το πολύ από τρεις καθηγητές)
- **Mathimata**
  - **isTeachedBy min 1 Teachers** (Ένα μάθημα διδάσκεται το λιγότερο από έναν δάσκαλο)
- **People**
  - **hasGender exaclty 1 Gender**(ενας ανθρωπος εχει ακριβως ενα φυλο)
- **Student**
  - **hasClass exactly 1 Taksi**(ενας μαθητης μπορει να ειναι φοιτα σε μια ταξη ανα ετος δεν μπορει πχ να ειναι ταυτοχρονα πρωτη και δευτερα)
  - **hasGrade exactly 1 Vathmologia**(ενας μαθητης εχει μια βαθμολογια στον ελεχγχο του)

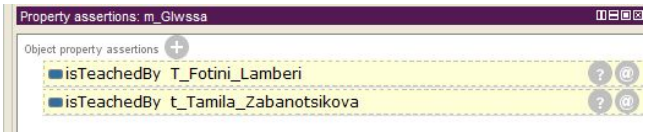
Reasoner

Είναι ανά κομμάτι λογισμικού το οποίο μας βοηθάει να εξάγουμε κάποια λογικά συμπεράσματα βάση τον συσχετίσεων που έχουμε κάνει.

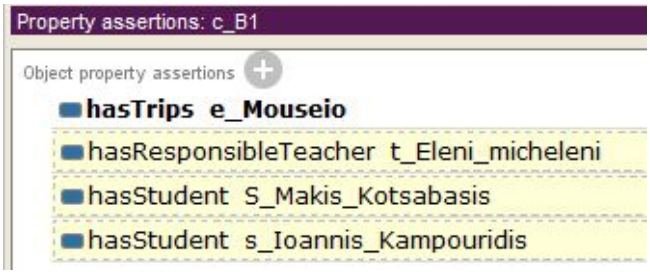
Όπως βλέπουμε στο protege όταν τρέχουμε τον reasoner

βγάζει κάποια επιπλέον συμπεράσματα χωρίς να τα έχουμε καταχωρήσει εμείς

Κάποια παραδείγματα αυτών:



Επειδή έχουμε βάλει στους 2 καθηγητές isTeaching την γλώσσα και έχουμε βάλει Inverse of το isTeaching με το TeachingBy , ο reasoner το αντιλαμβάνεται και βάζει αυτόματα στα individual του μαθήματος γλώσσας τα δεδομένα της παραπάνω φωτογραφίας.

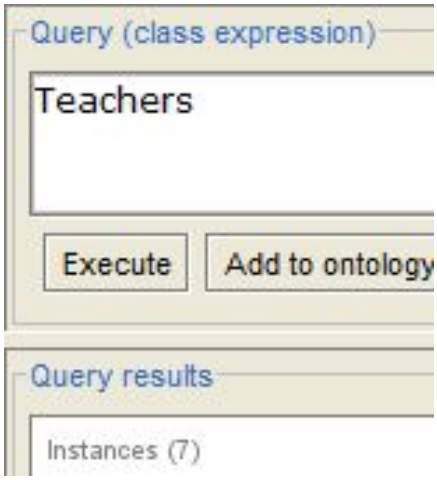


Παρομοίως σε αυτό το παράδειγμα επειδή το hasStudent το έχουμε βάλει inverseOf με το hasClass και ο Makis Kotsampasis και ο Ioannis kampouridis έχουν δηλωθεί ότι ανήκουν στο individual B1 , αυτόματα δημιουργείται στο individual B1 ότι έχει αυτούς τους δύο μαθητές.

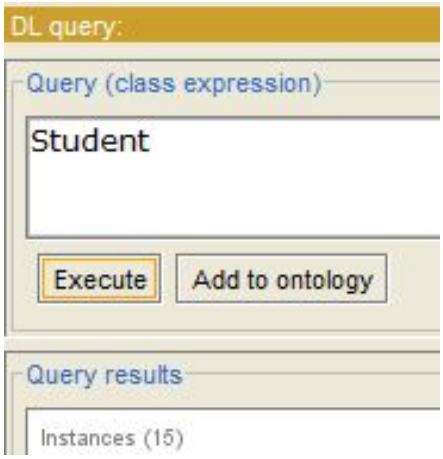
Επιπλέον στο ίδιο παράδειγμα με τον ίδιο τρόπο γίνεται το αντίστοιχο με το object property hasResponsibleTeacher.

Ερωτήματα DLQuery

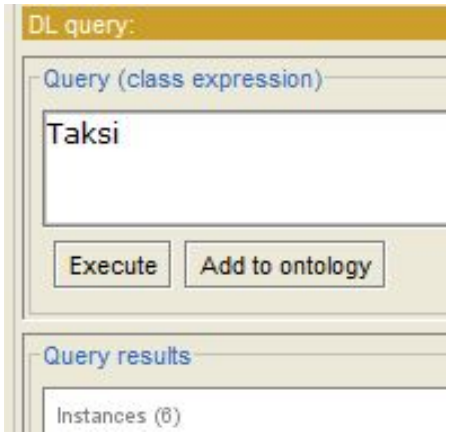
Στο πρώτο Dl Query Που τρέξαμε μας εμφανίζει όλους τους καθηγητές (7).



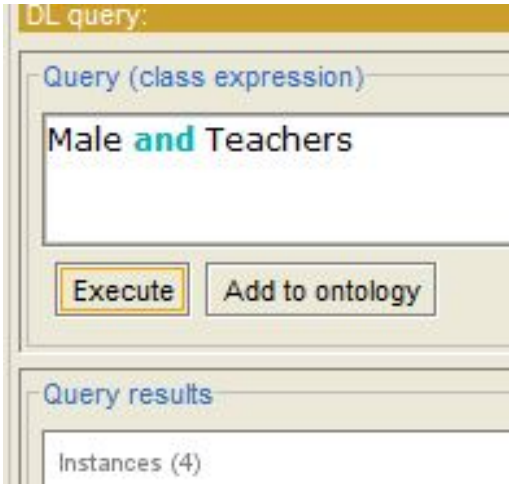
Εμφάνιση όλων των μαθητών(15)



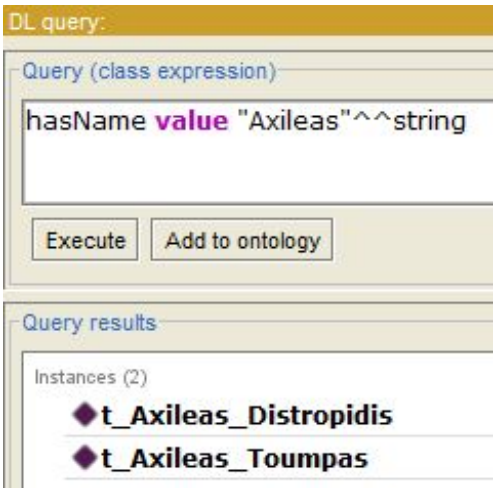
Εμφάνιση όλων των τμημάτων που υπάρχουν στο σχολείο(6)



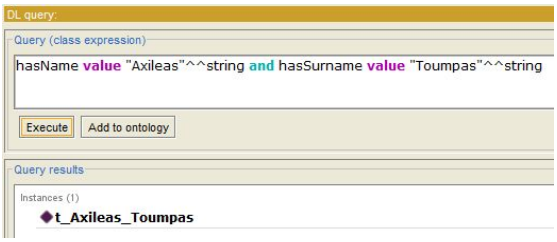
Εδώ εμφανίζονται όλοι οι καθηγητές.Δηλαδή από όλους τους καθηγητές εμφανίζονται μόνο οι άντρες και όχι οι γυναίκες.



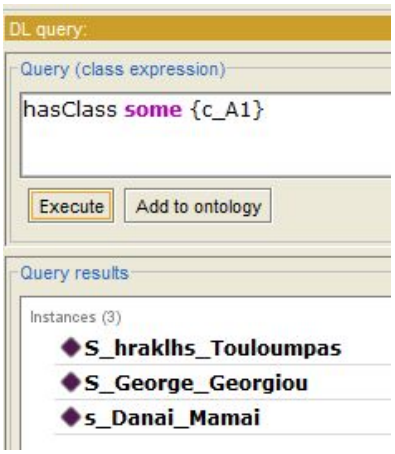
Εδώ εμφανίζουμε όλους τους ανθρώπους με όνομα Αχιλλέας.



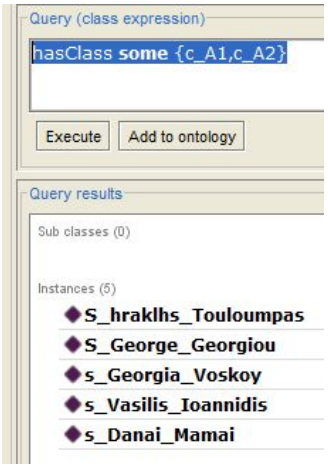
Στην επόμενη εικόνα εμφανίζουμε το άτομο με όνομα Αχιλλέας και επίθετο Τούμπας.



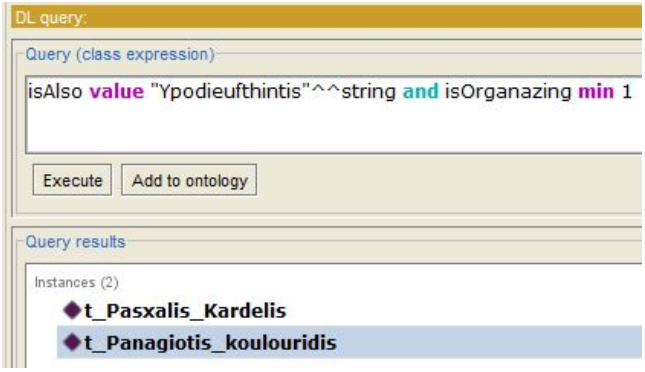
Εδώ βλέπουμε όλους τους μαθητές οι οποίοι ανήκουν στο τμήμα Α1.



Εδώ εμφανίζουμε τους μαθητές που ανήκουν γενικά στην πρώτη τάξη.



Εδώ εμφανίζουμε τα άτομα τα οποία είναι υποδιευθυντές και έχουν αναλάβει τουλάχιστον 1 εκδρομή.



Ερωτήματα SPARQL

1)Εμφάνιση των μαθημάτων που παρακολουθεί ο κάθε μαθητής.

```
SELECT ?hasName ?hasCourseName
WHERE
{
        ?x my:hasCourses ?z;
        my:hasName ?hasName.
        ?z my:hasCourseName ?hasCourseName.
}
order by ?hasCourseName
```

Το object property hasCourses έχει domain το Student(?x) και range το Mathimata(?z). Απο τον Student κρατάμε με το Data property hasName το όνομα του μαθητή που το αποθηκεύουμε στην μεταβλητή ?hasName και απο τα Mathimata κρατάμε με το hasCourseName το μάθημα που το αποθηκεύουμε στην μεταβλητή ?hasCourseName. Και έτσι με αυτόν τον τρόπο εμφανίζουμε όλα τα μαθήματα που παρακολουθεί ο κάθε μαθητής.

2)Εμφάνιση όλων που έχουν γεννηθεί από το 1970 έως το 2000.

```
SELECT ?x
WHERE
{
        ?x my:hasBirthYear ?y .
        FILTER (?y > 1970)
        FILTER (?y < 2000)
}
```

Το data property hasBirthYear έχει σαν domain το People(?x) και σαν range έχουμε βάλει ένα Integer(?y)(Χρονολογία γέννησης) Μέσω του Filter ελέγξαμε ποιοί από όλα τα άτομα έχουν γεννηθεί μετα το 1970 και πριν το 2000.Επομένως στο συγκεκριμένο παράδειγμα η μεταβλητή ?χ θα πάρει όλα τα individuals απο την κλάση People με την ηλικία που επιλέξαμε.

3.Εμφάνιση όλων εκτός των μαθητών.

```
SELECT ?x ?Surname
WHERE
{
        ?x rdfs:subClassOf my:People.
        ?y rdf:type ?x.
        MINUS {?y rdf:type my:Student}
        ?y my:hasSurname ?Surname.
}
```

Στη μεταβλητη(?χ) εμφανίζουμε τα περιεχόμενα της υποκλάσης People και το επώνυμο(?surname) μέσω του Data property hasSurname όπου εξαιρούνται (Minus) οι μαθητές .

4.Εμφάνιση όλων των βαθμίδων των καθηγητών, εάν έχουν.

```
SELECT ?eponimo  ?thesi
WHERE
{
        OPTIONAL {?y my:isAlso ?thesi  }
        ?y my:hasSurname ?eponimo.
        MINUS {?y rdf:type my:Student}
        MINUS {?y rdf:type my:Prosopiko}
        MINUS {?y rdf:type my:SullogosGonewm}
}
```

Εμφανίζουμε το επώνυμο(?eponimo) και την θέση(?thesi) όπου με το optional του λέμε ότι αν έχει κάποια θέση στο Is also να την εμφανίσει.Απο το data property παίρνουμε hasSurname δηλαδή το επώνυμο και εξαιρούμε(minus) τους Student,Prosopiko,SullogosGonewm .

5. Εμφάνιση όλων των τμημάτων της Β Λυκείου και πόσους μαθητές έχει το κάθε τμήμα.

```
SELECT ?tmima ?name (COUNT(?aName) AS ?hasStudent)
WHERE
 {
        ?tmima rdf:type my:Deutera.
        ?tmima my:hasTmhma ?name.
        ?aName my:hasClass ?tmima
 }
GROUP BY ?name ?tmima
ORDER BY ?tmima
```

Με (COUNT(?aName) AS ?hasStudent) κρατάμε σε μια μεταβλητή τον αριθμό των μαθητών που έχει κάθε τμήμα.Η υποκλάση του Tmima , Deutera δηλώνει ότι επιλέγουμε να εμφανίσουμε μόνο τα τμήματα της Β τάξης και το hasClass είναι το InverseOf του hasStudent.Το αποτέλεσμα του Query είναι ο αριθμός των μαθητων κάθε τμήματος της Β Λυκείου

6.Αριθμός μαθητών που πέτυχαν την αντίστοιχη βαθμολογία.

```
SELECT ?Vathmoi (COUNT(?u) AS ?count)
WHERE
{
        ?x  my:hasGrade ?z;
        ?u ?Vathmoi.
        ?Vathmoi rdf:type my:Vathmologia.
        ?z my:hasGradeSTR ?count
}
GROUP BY  ?Vathmoi
ORDER BY   ?count
```

Το object property hasGrade έχει σαν domain το Student(?x) και σαν range το Vathmologia(?z).

Στην μεταβλητή ?Vathmoi αποθηκεύουμε τα individuals των βαθμών τα οποία εχουν type την κλάση Vathmologia.

Μέσω του group by και του count βλέπουμε πόσα άτομα έχουν τον κάθε βαθμό.

REFERENCES

[1] Τάσεις και τεχνολογίες Σημασιολογικού Ιστού

[2] https://protege.stanford.edu/

[3] https://www.obitko.com/tutorials/ontologies-semantic-web/reasoning.html