ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Δημιουργία Πληροφοριακού Συστήματος Με Χρήση Οντολογιών Για Διαχείριση Εκδηλώσεων»

Του φοιτητή
Ιωάννη Ε. Άλλιου
Αρ. Μητρώου: 154428

Επιβλέπων
Ευκλείδης Κεραμόπουλος
Αναπληρωτής Καθηγητής

Ημερομηνία 12/06/2021

Τίτλος Δ.Ε.: Δημιουργία Πληροφοριακού Συστήματος με χρήση Οντολογιών για Διαχείριση Εκδηλώσεων

Κωδικός Δ.Ε. 19067

Ονοματεπώνυμο φοιτητή: Ιωάννης Ε. Άλλιος

Ονοματεπώνυμο εισηγητή: Ευκλείδης Κεραμόπουλος

Ημερομηνία ανάληψης Δ.Ε.: 23/12/2020

Ημερομηνία περάτωσης Δ.Ε.: 12/06/2021

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

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

«Στον πατέρα μου Ευάγγελο που με δίδαξε πως να βρίσκω

χρόνο για να στοχάζομαι την ουσία πραγμάτων.

... Πόσο θα ήθελα να είχα λίγο χρόνο ακόμα μαζί σου»

Πρόλογος

Η παρούσα Διπλωματική Εργασία (ΔΕ) εκπονήθηκε στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων της Σχολής Μηχανικών του Διεθνούς Πανεπιστήμιου της Ελλάδας (ΔΙΠΑΕ)

Έχοντας ολοκληρώσει το βασικό πρόγραμμα σπουδών και επιλέγοντας μαθήματα του τομέα της πληροφορικής ήρθα σε επαφή με διάφορες τεχνολογίες, γνώριζα ήδη την ύπαρξη μερικών και κάποιες από αυτές τις είχα δει σε χρήση κατά τον επαγγελματικό μου βίο. Οι τεχνολογίες όμως που συνάντησα για πρώτη φορά άνοιξαν έναν νέο ορίζοντα στα ενδιαφέροντα μου και μου αποκάλυψαν τομείς έρευνας και πειραματισμού που δεν μπορούσα να φανταστώ. Σαν φιλοσοφία ζωής έχω την συνεχή μάθηση και βελτίωση, αισθάνομαι πραγματικά τυχερός που συνάντησα και διδάχτηκα από τα στελέχη της Σχολής Μηχανικών του ΔΙΠΑΕ τα οποία μεταλαμπαδεύουν όχι μόνο την γνώση αλλά και την νοοτροπία της συνεχούς έρευνας και της μάθησης μέσω του πειραματισμού.

Σε αυτό το πλαίσιο σκέψης θεώρησα ότι ο Σημασιολογικός Ιστός έχει να μου προσφέρει νέες γνώσεις και ένα ενδιαφέρον πεδίο έρευνας, επίσης η ενασχόληση με την Οντολογική διαχείριση της πληροφορίας απαιτεί γνώσεις από μεγάλο ποσοστό, αγαπημένων, μαθημάτων που παρακολούθησα στην σχολή.

Το θέμα της ΔΕ, πέρα από την Οντολογική του διάσταση, ζητά την ανάπτυξη ενός Πληροφοριακού Συστήματος, πράγμα που σημαίνει δίδεται η δυνατότητα να σχεδιαστεί κάθετα αυτό το ΠΣ εκθέτοντας με σε όλες τις ανάγκες της δημιουργίας του.

Περίληψη

Σε αυτή την Διπλωματική Εργασία εξετάσθηκε η δυνατότητα της δημιουργίας ενός Πληροφοριακού Συστήματος διαχείρισης εκδηλώσεων με χρήση Οντολογιών. Περιορισμός υπάρχει ως προς τον τρόπο αποθήκευσης της πληροφορίας η οποία πρέπει να γίνει σε owl, αλλά και στον τρόπο άντλησης της πληροφορίας η οποία πρέπει να γίνει με την χρήση SPARQL.

Αρχικά ορίσθηκε το τι θεωρείται Εκδήλωση και ποιες πληροφορίες είναι αυτές που θα έπρεπε να παρέχει το ΠΣ προσέχοντας να είναι αρκετά γενικευμένο στην χρήση του, αλλά και να δίνει την δυνατότητα να είναι και εξειδικευμένο αν αυτό αποφασισθεί από τον χρήστη.

Έγινε κάθετη σχεδίαση της εφαρμογής και καθορίστηκαν οι απαιτήσεις ενός τέτοιου ΠΣ χωρίς να ληφθούν υπόψη οι τυχών ιδιαιτερότητες της χρήσης Οντολογιών και SPARQL αλλά και των περιρρέοντων περιορισμών ή οφελών τους.

Παρουσιάζεται UML με τις κλάσεις, τα δεδομένα και τις σχέσεις τους.

Εξετάζονται, βιβλιογραφικά, οι διαθέσιμες τεχνολογίες για την ανάπτυξη του ΠΣ, έγινε επιλογή μερικών και απόρριψη άλλων βάσει της αποδοτικότητας αλλά και της ύπαρξης προηγούμενης εμπειρίας.

Σχεδιάστηκε και δοκιμάστηκε η οντολογία με την χρήση του περιβάλλοντος Protégé, καταχωρήθηκαν πειραματικά δεδομένα, τα οποία ανέβηκαν στην πλατφόρμα Open Source Virtuoso.

Έγινε τροποποίηση της αρχικής σχεδίασης λαμβάνοντας υπόψη την χρήση Οντολογιών, πάρθηκαν αποφάσεις σε ότι αφορά την μοναδικότητα των κλειδιών και την υλοποίηση του ΠΣ με τέτοιο τρόπο ώστε η αποθηκευμένη (εξαγόμενη) πληροφορία να είναι ένας οντολογικά γράφος, που μπορεί να υπάρξει και να φέρει σωστά αποτελέσματα και εκτός του ΠΣ.

Σχεδιάστηκαν οι διεπαφές χρήστη με το εργαλείο Scene Builder.

Έγινε πειραματισμός σε SPARQL απέναντι την πλατφόρμα Virtuoso και αποφασίστηκε η δομή των ερωτημάτων SPARQL για αναζήτηση, καταχώρηση και αλλαγή, που θα χρησιμοποιηθεί στο ΠΣ

Αναλύεται βήμα-βήμα όλη η ανάπτυξη του ΠΣ, περιγράφονται οι λειτουργίες του και γίνεται έλεγχος ως προς την ορθότητα του παραγόμενου γράφου με τα κατάλληλα εργαλεία πιστοποίησης.

Αναλύονται οι δυσκολίες, οι περιορισμοί και οι αποφάσεις που λήφθηκαν σε σχέση με τεχνολογίες που δεν χρησιμοποιήθηκαν.

Προτείνονται βελτιώσεις που θα μπορούσαν να γίνουν στο μέλλον.

«Creating an Event Management Information System using Ontologies»

Ioannis E. Allios

Abstract

This Diploma Thesis examines the possibility of creating an Event Management Information System using Ontologies.

There are restrictions on how to store information, which must be done in owl, but also on how to retrieve information, that must be done using SPARQL. Initially, it is defined what is considered an Event and what kind of information should be provided, taking care to be quite generalized in its use, but also to be able to specialize if this is decided by the user.

The application was designed vertically and the requirements of such System were determined without taking into account, any peculiarities of the use of Ontologies and SPARQL, also their surrounding limitations or benefits. A UML design with classes, data and their relationships is presented.
A literature survey is taken on the available technologies for the development of such system, some technologies were selected based on the efficiency but also the existence of previous experience.
The ontology was designed and tested using the Protégé environment, experimental data were entered, which were uploaded to the Open Source Virtuoso platform.
The original design was modified taking into account the use of Ontologies, decisions were made regarding the uniqueness of the key data and the implementation of the application in such a way that the stored (exported) information is an valid ontological graph, which can exist and provide correct results in any Ontological environment. User interfaces were designed with the Scene Builder tool.
SPARQL queries were experimented on the ihu.gr's Virtuoso platform. Based on these results, the structure of the SPARQL queries for search, insert and update was decided, which will be used in the application.
The entire development of the application is analyzed step by step, its functions are described and the correctness of the generated graph is checked with the appropriate certification tools.
The difficulties, limitations and decisions made are presented in relation to technologies that were not used are analyzed.
Improvements are suggested that could be made in the future

Ευχαριστίες

Θέλω να ευχαριστήσω τον Καθηγητή **Ευκλείδη Κεραμόπουλο** για τις γνώσεις που μου προσέφερε καθ' όλη την διάρκεια των σπουδών μου στις Βάσεις Δεδομένων, Διεπαφές Χρήστη, Σημασιολογικό Ιστό και Java, αλλά και για την ευελιξία, χρόνου και επιλογών υλοποίησης, που μου επέτρεψε.

Την σύζυγο μου **Βασιλική Μπρόζου**, που με στήριξε σε όλα τα επίπεδα.

Τις κόρες μου **Δέσποινα Άλλιου** και **Ραφαέλα Άλλιου** που με κατανόηση δέχθηκαν την απουσία μου σε κάποιες δραστηριότητες τους.

Τον καλό φίλο Μηχανικό Πληροφορικής **Μιχάλη Καρυπίδη** που πάντα με ωθούσε να προχωράω, όταν εγώ σταματούσα.

Την Προϊσταμένη Διεύθυνσης ΕΟΠΥΥ Κιλκίς **Ελισάβετ Φρεγγίδου** που, υπηρεσιακά αλλά και ανθρώπινα, μερίμνησε ώστε να μην χάσω κανένα εργαστήριο και καμιά εξέταση καθ' όλη την διάρκεια των σπουδών μου.

Περιεχόμενα

Πρόλογος v
Περίληψη vi
Abstract vii
Ευχαριστίες ix
Περιεχόμενα x
Κατάλογος Σχημάτων xiii
Κατάλογος Πινάκων xiv
Συντομογραφίες xv
Κεφάλαιο 1ο: Εκδήλωση σαν πληροφορία 1
1.1 Εισαγωγή 1
1.2 Ορολογία 1
1.3 Σημασία και δομή 2
1.4 Χαρακτηριστικά 3
1.5 Συστατικά της εκδήλωσης 3
1.6 Υπόθεση εξειδίκευσης 4
1.7 Υπόθεση ομαδοποίησης 5
1.8 Μοντελοποίηση πληροφορίας 6
1.8.1 Πιθανές Κλάσεις και Πληροφορίες 6
1.8.2 Ειδίκευση, πληθυσμός και σύνθεση κλάσεων 7
1.9 Μοντελοποίηση UML 9
1.9.1 Umbrello UML Modeler 9
1.9.2 Γραφική αναπαράσταση κλάσεων και σχέσεων 9
1.10 Συμπεράσματα πριν την υλοποίηση 10
Κεφάλαιο 2ο: Τεχνολογίες υλοποίησης 11
2.1 Τεχνολογικές προϋποθέσεις 11
2.2 Η γλώσσα προγραμματισμού Java 12
2.3 Δομές δεδομένων και πληροφορίας 13
2.3.1 Extensible Markup Language 13
2.3.2 Resource Description Framework 16
2.3.3 RDF Schema 17
2.4 Σημασιολογικός Ιστός 18
2.4.1 Αρχιτεκτονική 19
2.4.2 Το μέλλον του Σ.Ι. 20

2.5 Οντολογία....20
2.5.1 Συστατικά της οντολογίας....20
2.5.2 Web Ontology Language....22
2.5.3 Δομή της οντολογίας....23
2.6 Γλώσσα SPARQL....23
2.6.1 Γενικά....23
2.6.2 Σύνταξη Turtle....24
2.6.3 Δομή SPARQL....25
2.6.4 Ερώτημα SPARQL....26
2.6.5 SPARQL Update, Delete, Insert....28
2.7 Βιβλιοθήκη Apache Jena....29
2.7.1 RDF API....30
2.7.2 Ontology API....30
2.7.3 Μηχανισμός ARQ....30
2.7.4 Fuseki server....31
2.8 JavaFX....31
2.8.1 FXML....31
2.8.2 Scene Builder....32
Κεφάλαιο 3ο: Υλοποίηση συστήματος....33
3.1 Δημιουργία οντολογίας....33
3.1.1 Τροποποίηση κλάσεων και σχέσεων....33
3.1.2 Υλοποίηση κλάσεων και Data properties στο Protégé....35
3.1.3 Υλοποίηση Object Properties στο Protégé....39
3.1.4 Επισκόπηση οντολογίας....41
3.2 Έλεγχος οντολογίας....41
3.2.1 Καταχώρηση Individual....42
3.2.2 Protégé και reasoner....42
3.3 S PARQL Query....45
3.3.1 Protégé SPARQL Queries....45
3.3.2 Virtuoso endpoint....47
3.3.3 OpenLink Virtuoso Graph....47
3.3.4 Δοκιμή Query....50
3.3.5 Ενεργοποίηση Update σε endpoint....51
3.4 Δημιουργία εφαρμογής....52
3.4.1 Περιβάλλον και Βιβλιοθήκη....52

3.4.2 Έλεγχος καταχώρησης δεδομένων 53
3.4.3 Jena και πρόσβαση στη πληροφορία 54
3.4.4 Περιγραφή βοηθητικών συναρτήσεων 57
3.4.5 Γενικό Διάγραμμα ροής αρθρώματος FXML 61
3.4.6 Διάγραμμα ροής επικοινωνίας με την οντολογία 62
3.4.7 Παράδειγμα υλοποίησης 62
3.4.8 Παράδειγμα Update 65
3.4.9 Παράδειγμα καταχώρησης νέας κλάσης 69
3.5 Νέα δεδομένα ελέγχου ΠΣ 70
3.6 Εισαγωγή δεδομένων 72
3.7 Εξαγωγή δεδομένων παρατήρηση σύνταξης 77
Κεφάλαιο 4ο: Συμπεράσματα 79
4.1 Γενικά 79
4.2 Έλεγχος στο Protégé 79
4.3 Προβλήματα 81
4.4 Μελλοντική υλοποίηση 81
4.5 Στόχος επόμενης σχετικής έρευνας 81
ΒΙΒΛΙΟΓΡΑΦΙΑ 83
ΠΑΡΑΡΤΗΜΑ Α Κώδικας Turtle Οντολογίας 87
ΠΑΡΑΡΤΗΜΑ Β Κώδικας Turtle Individuals 99
ΠΑΡΑΡΤΗΜΑ C Java JavaFX FXML 109
ΠΑΡΑΡΤΗΜΑ D Στιγμιότυπα οθόνης 129

Κατάλογος Σχημάτων

Σχήμα 1.1: Αρχική υλοποίηση UML....10
Σχήμα 3.1: XML and Semantic Web W3C Standards Timeline-History....16
Σχήμα 2.2: Validation από το www.w3.org....17
Σχήμα 2.3: The Semantic Web Stack....19
Εικόνα 2.4: JavaFX Scene Builder....32
Σχήμα 3.1: Διάγραμμα Πληροφοριακού Συστήματος....33
Σχήμα 3.2: Διάγραμμα κλάσης Entity....35
Σχήμα 3.3: Διάγραμμα κλάσης Participant....35
Σχήμα 3.4: Διάγραμμα κλάσης Place....36
Σχήμα 3.5: Διάγραμμα κλάσης Product....37
Σχήμα 3.6: Διάγραμμα κλάσης Happening....37
Σχήμα 3.7: Διάγραμμα κλάσης Scope....38
Σχήμα 3.8: Διάγραμμα κλάσης Event....38
Σχήμα 3.9: Διάγραμμα σχέσεων Happening - Entity....39
Σχήμα 3.10: Διάγραμμα σχέσεων Happening - Event....39
Σχήμα 3.11: Διάγραμμα σχέσεων Product - Event....40
Σχήμα 3.12: Διάγραμμα σχέσεων Event – Place & Event – Entity....40
Σχήμα 3.13: OntoGraph από Protege....41
Εικόνα 3.14: Protégé IND-ENT-PY-03....43
Σχήμα 3.15: Protégé OntoGraph IND-EVENT-01....44
Εικόνα 3.16: Protégé IND-HAPP-01 συμπερασμός....45
Εικόνα 3.17: Protégé ερώτημα 1....45
Εικόνα 3.18: Protégé ερώτημα 2....46
Εικόνα 3.19: Protégé ερώτημα 3....46
Εικόνα 3.20: ~Okeaos, Ubuntu Server VM....47
Εικόνα 3.21: Virtuoso Conductor Quad Store Upload....48
Εικόνα 3.22: Virtuoso Conductor SPARQL ερώτημα 3....49
Εικόνα 3.23: Virtuoso endpoint SPARQL ερώτημα 3....50
Εικόνα 3.24: Αποτέλεσμα endpoint σε μορφή HTML....50
Εικόνα 3.25: Endpoint χωρίς δυνατότητα update....51
Εικόνα 3.26: Conductor, User Accounts....51
Εικόνα 3.27: Conductor, Edit User Account....52
Σχήμα 3.28: Γενικό διάγραμμα ροής αρθρώματος....61
Σχήμα 3.29: Διάγραμμα ροής επικοινωνίας με οντολογία....62
Εικόνα 3.30: Αρχική οθόνη ΠΣ....63
Εικόνα 3.31: Δήλωση VBox στο splash2.fxml....63
Εικόνα 3.32: Menu επιλογής....64
Εικόνα 3.33: Άρθρωμα Καταχώρηση νέου συμβάντος....65
Εικόνα 3.34: URI έλεγχος....66
Εικόνα 3.35: UI Choice Box «Συμμετέχει Ως»....67
Εικόνα 3.36: UI Choice Box άτομο....68
Εικόνα 3.37: UI List View άτομο σε συμβάν....68
Εικόνα 3.38: UI Καταχώρηση νέας κατηγορίας παραγώγου εκδήλωσης....70
Εικόνα 3.39: UI Καταχώρηση νέας κατηγορίας και κώδικας Turtle....72

Εικόνα 3.40: UI Καταχώρηση νέου Individual και κώδικας Turtle....73
Εικόνα 3.41: UI Καταχώρηση νέου Individual Entity και κώδικας Turtle....73
Εικόνα 3.42: UI Καταχώρηση νέου Individual Entity και κώδικας Turtle....74
Εικόνα 3.43: UI Καταχώρηση νέου Individual Happening και κώδικας Turtle....75
Εικόνα 3.44: UI Καταχώρηση νέου Individual Event και κώδικας Turtle....76
Εικόνα 3.45: UI Εξαγωγή οντολογίας....77
Εικόνα 4.1: Protégé Test Event Individual....79
Εικόνα 4.2: Protégé Test Happening Individual....79
Εικόνα 4.3: Protégé Test Entity Individual....80
Εικόνα 4.4: Protégé Κλάσεις δηλωμένες από χρήστη....80

Κατάλογος Πινάκων

Πίνακας 2.1: Triplets of the Data Model....17
Πίνακας 2.2: Βασικά πακέτα Jena....29
Πίνακας 3.1: Παράδειγμα ονομασίας Individual....42

Συντομογραφίες

ΔΕ	Διπλωματική Εργασία
ΜΛ	Μηχανική Λογισμικού
ΔΙΠΑΕ	Διεθνές Πανεπιστήμιο Ελλάδος
ΠΣ	Πληροφοριακό Σύστημα
ΛΣ	Λειτουργικό Σύστημα
UI	User Interface
DOM	Document Object Model

Κεφάλαιο 1ο: Εκδήλωση σαν πληροφορία

1.1 Εισαγωγή

Στην επιστήμη της ανάπτυξης λογισμικού συχνά δεν λαμβάνεται υπόψη ως πεδίο ορισμού ο πραγματικός κόσμος, η σημασία των λέξεων, η σημασία των εννοιών και το πλαίσιο εφαρμογής τους σαν νόημα. Αποτέλεσμα είναι, συνήθως, από την μία πλευρά να έχουμε καλά ορισμένες συναρτήσεις, όμορφο κώδικα, εκτενείς τεκμηρίωση και τελευταίας λέξης διεπαφές, αλλά από την άλλη πλευρά να υπάρχουν τεράστια κενά στην κατανόηση αφηρημένων εννοιών που είναι μεν αυτονόητες για τον χρήστη του λογισμικού, αποτελούν δε "*μυστηριώδεις απαιτήσεις των χρηστών*" για τον δημιουργό και συντηρητή ενός ΠΣ.

Η αφηρημένη περιγραφή θα πρέπει να μοντελοποιείται με τρόπο που να ενώσει τους δύο κόσμους, τον φυσικό και αυτόν της καταγραφής του σαν πληροφορία, πρακτικά η μετάπτωση από το υπαρκτό στο περιγραφικό σημαίνει απώλεια ακρίβειας, ίσως και στρέβλωση σημασίας. Αυτό είναι ένα θέμα που απασχόλησε και θα συνεχίσει να απασχολεί τους επαγγελματίες της ΜΛ.

1.2 Ορολογία

Αναζητήθηκε ο όρος **"Εκδήλωση" [1]** στα ελληνικά. Αυτό ορίζεται σαν:

«*η ενέργεια ή το αποτέλεσμα του εκδηλώνω· συμπεριφορά, πράξη, δραστηριότητα που φανερώνει, αποκαλύπτει μια ψυχική ή διανοητική κατάσταση ή διάθεση: Εκδηλώσεις χαράς / λύπης / οργής. Αυθόρμητη ~. ~ ενδιαφέροντος. 2. ομαδική δημόσια πράξη με την οποία επιδιώκεται η έκφραση άποψης, στάσης, διάθεσης κτλ.: Πολιτική / καλλιτεχνική ~. Εορταστικές / επετειακές / πανηγυρικές εκδηλώσεις. Εκδηλώσεις διαμαρτυρίας / συμπαράστασης. Ειρηνικές / βίαιες εκδηλώσεις. Η έναρξη των εκδηλώσεων θα γίνει μεθαύριο.*».

<< εκδηλω-(δες εκδηλώνω) -ση >>, και μας οδηγεί στο "Εκδηλώνω" που είναι << ἐκδηλ(ῶ) `δείχνω καθαρά -ώνω κάτι>>.

Αναζητήθηκε ο Αγγλικός **[2]** όρος "Event" δίδεται:

«something that happens or is regarded as happening; an occurrence, especially one of some importance».

Αναζητήθηκε ο Λατινικός όρος "Event"**[3]** και βρέθηκε να έχει ρίζα το "Evenire" με την σημασία «αποτέλεσμα» και «συνέβη».

Και στις δύο γλώσσες υπάρχει το κοινό χαρακτηριστικό μιας «Υπαρκτής Κατάστασης» που έχει σημασία για κάποιους και μπορεί να παρατηρηθεί ή να (ανα)**δειχθεί**. Η σύνδεση μεταξύ του "δείχνω καθαρά" και του "Event" είναι "το συμβάν" ή "το αποτέλεσμα".

Για τις ανάγκες της παρούσας Διπλωματικής Εργασίας θα θεωρηθεί ο όρος **«Εκδήλωση» ως ταυτόσημο του όρου «Event»** και ο όρος **«Συμβάν» ως ταυτόσημο του όρου «Happening»**, περιορίζοντας τον ευρύ και πολυχρηστικό ελληνικό ορισμό σαν ίσο με αυτόν, της σημερινής Lingua Franca, των αγγλικών.

1.3 Σημασία και δομή

Εκδήλωση είναι η διοργάνωση μίας δημόσιας συνεργατικής πράξης ή δραστηριότητας για ποικίλους σκοπούς, εορταστικούς, διαμαρτυρίας, επετειακούς, εμπορικούς κ.α. Επιλέχθηκε να μην συμπεριληφθούν εκδηλώσεις κοινωνικό-παραδοσιακού χαρακτήρα όπως γάμοι, βαφτίσεις, σουνέτι, μπαρ μιτζβά, κτλ και να επικεντρωθεί η μελέτη σε σύγχρονες εμπορικό-κοινωνικές εκδηλώσεις "Δυτικού" τύπου όπως συνέδρια, βραβεύσεις, παρουσιάσεις, ημερίδες, ανοιχτές ομιλίες κτλ. Αναζητήθηκε ένας ορισμός από την επιστημονική φαρέτρα του τομέα της Διοίκησης και Management. Με πρακτική, πλέον, σκέψη διαπιστώνουμε ότι μια εκδήλωση υλοποιείται με τον κόπο πολλών ανθρώπων, που θα παίξουν έναν ρόλο ο κάθε ένας, ο ρόλος αυτός είναι οι **Οργανικές Θέσεις της.[4]**
η διοίκηση της εκδήλωσης,

- τα οικονομικά της,
- η γραμματειακή υποστήριξη,
- η διεύθυνση και η διαχείριση ανθρώπινου δυναμικού,
- το marketing,
- η συντήρηση και η τεχνική υποστήριξη,
- η τροφοδοσία,
- η ασφάλεια,
- η εξυπηρέτηση πελατών, κ.α.

Η συστηματική υλοποίηση εκδηλώσεων απαιτεί: Ειδικευμένους επαγγελματίες που ασχολούνται μόνο με αυτές, ειδικευμένες επιχειρήσεις παροχής υπηρεσιών, ειδικά διαμορφωμένους χώρους (φυσικούς ή εικονικούς). Η χρήση και ο τρόπος χρήσης των πόρων είναι επιλογές που πρέπει να κάνει ο Manager μιας εκδήλωσης γιατί θα πρέπει να φροντίσει για **[5]**:

- τη μελέτη και ταξινόμηση του χώρου
- τον προγραμματισμό-Αναλυτικό σχεδιασμό
- τη διακόσμηση
- τον εξοπλισμό-τεχνικό σχεδιασμό
- τον σχεδιασμό οικονομικού πλάνου Cashflow management
- την οπτικοακουστική μελέτη χώρου & φωτισμό
- την υγεία και ασφάλεια του κοινού
- τις χορηγίες- τους σπόνσορες
- τα εισιτήρια και τις προσκλήσεις
- την προώθηση-διαφήμιση
- το Information desk
- το προσωπικό εξυπηρέτησης πελατών
- τη μεταφορά και διαμονή κοινού
- το παρκινγκ για τα αυτοκίνητα
- εστίαση catering
- τα αναμνηστικά δώρα

Αν όμως μία οντότητα, όπως μια επιχείρηση ή σύλλογος, κάνουν εκδηλώσεις σποραδικά τότε δεν υπάρχει ανθρώπινο δυναμικό που ασχολείται αποκλειστικά με τις ανάγκες της εκδήλωσης αλλά οι παραπάνω οργανικές θέσεις (και ο ρόλος του Manager) καλύπτονται από το υπάρχον προσωπικό της οντότητας και από εθελοντές.

1.4 Χαρακτηριστικά

Εύκολα διαπιστώνεται ότι οι εκδηλώσεις μπορούν να διαφέρουν πάρα πολύ μεταξύ τους, για παράδειγμα οι παρακάτω εκδηλώσεις:

- Η Παφλαγονία σε αρχαία και ιστορικά κείμενα
- Τεχνικές επιβίωσης κρατουμένου σε έδαφος που δεν ισχύει η συνθήκη της Γενεύης
- Uwe Boll, Director and producer of “Auschwitz”, QnA

Το πραγματευόμενο θέμα των παραπάνω εκδηλώσεων ίσως να θεωρηθεί ως άκρον άωτον μιας κατάστασης, για κάποιους οι οποίοι δεν εμπλέκονται ενεργά, με το ίδιο το θέμα της εκδήλωσης.

Παρά την κραυγαλέα θεματική απόκλιση υπάρχει απόλυτη σύγκλιση σε ορισμένα πρακτικά θέματα που τίθενται ως εύλογα ερωτήματα:

- Ποιός την κάνει;
- Ποιος είναι αυτός που την οργανώνει;
- Πού θα γίνει;
- Πότε Θα γίνει;
- Πόσο διαρκεί;
- Ποιοι θα συμμετέχουν;
- Ποιοι είναι αυτοί που συμμετέχουν;
- Τι θα γίνει στην εκδήλωση;
- Που θα διαβάσω για αυτήν;
- Θα υπάρξει βιντεοσκόπηση; Κοκ.

Φυσικά τα ερωτήματα αυτά μπορούν να διατυπωθούν και σε αόριστο χρόνο για εκδήλωση που έχει συμβεί ήδη.

Η απάντηση αυτών των ερωτημάτων δίδει και τα χαρακτηριστικά μιας εκδήλωσης που ισχύουν στις περισσότερες περιπτώσεις.

Περαιτέρω ανάλυση της δομής και των αναγκών μιας εκδήλωσης, θεωρήθηκε ότι, ξεφεύγουν από τα ενδιαφέροντα του αναγνώστη, που στην προκειμένη περίπτωση είναι Μηχανικός ή επαγγελματίας του τομέα της Πληροφορικής.

1.5 Συστατικά της εκδήλωσης

Όταν έχει επιλεχθεί, από ένα υπαρκτό σύστημα, η πληροφορία που χρειάζεται να διαχειρίζεται ένα ΠΣ, γίνεται μια αρχική οργάνωση της δομής της. Συγκεκριμένα επιλέγονται τα συστατικά της, γίνονται υποθέσεις για διάφορες πτυχές της, αναλύεται αν τα επιλεγμένα συστατικά καλύπτουν τις πιθανές ανάγκες και αναπαριστάται με έναν δομημένο και τυπικό τρόπο η πληροφορία.

Επιλέχθηκε από την πληθώρα των οργανικών συστατικών μιας εκδήλωσης ένα μέρος αυτών, το οποίο θεωρήθηκε βασικό και ουσιώδες.
Αφηρημένα (Abstract) ορίζεται ότι μία εκδήλωση σίγουρα έχει :

- Διοργανωτή που είναι φυσικό ή νομικό πρόσωπο
- Χώρο διεξαγωγής που είναι γεωγραφικός ή εικονικός ή και τα δύο
- Χρόνο διεξαγωγής, ημερομηνίες αρχής και τέλους

- Σκοπό σαν στόχο που την καθορίζει προς τον έξω κόσμο
- Συμβάντα που πραγματοποιούνται στη διάρκεια της εκδήλωσης
- Συμμετέχοντες, που είναι φυσικά ή νομικά πρόσωπα, στα συμβάντα
- Οι συμμετέχοντες έχουν ένα ή περισσότερους ρόλους στο συμβάν
- Οι ρόλοι είναι σχετικοί με τον σκοπό ή απλά λειτουργικοί
- Προϊόν παράγωγο που προκύπτει ή προηγείται της εκδήλωσης

Τα παραπάνω συστατικά βρίσκονται σε κάθε εκδήλωση ανεξαρτήτως του κυρίως θέματος της. Προχωρώντας σε εξειδίκευση των αφηρημένων αυτών εννοιών καλύπτονται όλες οι πληροφοριακές ανάγκες της, πάντα στα πλαίσια των χαρακτηριστικών που στο προηγούμενο κεφάλαιο αναφέρθηκαν.

1.6 Υπόθεση εξειδίκευσης

Από την αφηρημένη επιλογή συστατικών που έγινε, δοκιμάζονται υποθέσεις (περιπτώσεις) πληροφορίας με σκοπό να αποκαλυφθεί, συστατικό που είναι ουσιώδες αλλά δεν επιλέχθηκε στην παράγραφο 1.5

Υπόθεση «*Εταιρεία φύλαξης*» που έχει αναλάβει την φύλαξη της εκδήλωσης, πώς θα μπορούσε να περιγραφεί;

- Είναι νομικό πρόσωπο
- Συμμετέχει σε μια εκδήλωση
- Έχει ρόλο λειτουργικό

Υπόθεση «*Μπροσούρα φυλλάδιο*» που έχει εκδοθεί για προώθηση της εκδήλωσης και ενημέρωση του κοινού για τον σκοπό και τους συμμετέχοντες της, πώς περιγράφεται;

- Αφορά μία και μόνο εκδήλωση
- Είναι παράγωγο προϊόν που προηγείται της εκδήλωσης

Υπόθεση «*Τεχνικός φωτισμού*», πώς περιγράφεται;

- Είναι φυσικό πρόσωπο
- Συμμετέχει σε μια εκδήλωση
- Έχει λειτουργικό ρόλο

Υπόθεση «*Απονομή βραβείων»,* πώς περιγράφεται;

- Είναι συμβάν
- Αποτελεί μέρος μια εκδήλωσης
- Αφορά 1 έως πολλά φυσικά ή νομικά πρόσωπα
- Καθορίζει ρόλους σε φυσικά ή νομικά πρόσωπα (Βραβευόμενος, Ομιλητής, φορέας που βραβεύει τις οντότητες)
- Προκαλεί παράγωγο εκδήλωσης (Βραβείο, Φωτογραφίες, Δημοσιεύσεις)

Διαπιστώθηκε ότι η αφηρημένη περιγραφή της πληροφορίας που έγινε στο κεφάλαιο 1,5, εξειδικεύεται χωρίς προβλήματα, τουλάχιστον σε αυτή την φάση της έρευνας.

1.7 Υπόθεση ομαδοποίησης

Πέραν των αυτονόητων άμεσων ομαδοποιήσεων, θα μπορεί κάποιος να δει βάσει των επιλεγμένων συστατικών, ομάδες πληροφορίας από στοιχεία που δεν σχετίζονται ευθέως;

Υπόθεση «*Συμμετέχοντες Εκδήλωσης*» πώς θα μπορούσε να περιγραφεί αυτό το σύνολο;

- Η εκδήλωση έχει συμβάντα
- Το κάθε συμβάν έχει συμμετέχοντες

Προκύπτει έμμεση συσχέτιση της εκδήλωσης με τους συμμετέχοντες μέσω του συνόλου των συμβάντων της εκδήλωσης

Υπόθεση «*Χώρος με Εκδήλωση που έχει Ημερομηνία έναρξης Χ*» πώς θα μπορούσε να περιγραφεί αυτό το σύνολο;

- Η εκδήλωση έχει έναν χώρο πραγματοποίησης
- Ο χώρος φιλοξενεί από καμία έως και το σύνολο των εκδηλώσεων
- Η εκδήλωση έχει ημερομηνία έναρξης

Προκύπτει άμεση συσχέτιση του χώρου με την εκδήλωση, άμεση συσχέτιση της ημερομηνίας Χ με την εκδήλωση, έμμεση συσχέτιση του χώρου με την ημερομηνία Χ

Υπόθεση «*Σε ποιες εκδηλώσεις συμμετείχε το Χ άτομο;*» πώς θα μπορούσε να περιγραφεί αυτό το σύνολο;

- Το άτομο Χ συμμετέχει σε κανένα έως και όλα τα συμβάντα μιας εκδήλωσης
- Το συμβάν μιας εκδήλωσης είναι διαφορετικό από το συμβάν μιας άλλης εκδήλωσης ακόμα και αν οι συμμετέχοντες των συμβάντων και οι ρόλοι τους είναι ίδιοι
- Η κάθε εκδήλωση αποτελείται από ένα έως και όλα τα συμβάντα

Προκύπτει έμμεση συσχέτιση του Χ με τις εκδηλώσεις που περιλαμβάνουν συμβάντα με συμμετέχοντα τον Χ.

Υπόθεση «*Ποια άτομα συμμετείχαν σε εκδηλώσεις Σκοπού Χ*» πώς θα μπορούσε να περιγραφεί αυτό το σύνολο;

- Η κάθε εκδήλωση έχει έναν και μόνο σκοπό
- Ο κάθε σκοπός έχει από καμία έως και όλες τις εκδηλώσεις
- Άτομα συμμετέχουν σε συμβάντα
- Η κάθε εκδήλωση αποτελείται από ένα έως και όλα τα συμβάντα

Προκύπτει έμμεση συσχέτιση του των οντοτήτων με τις εκδηλώσεις και περιορισμός μέσω άμεσης συσχέτισης των εκδηλώσεων με τον σκοπό Χ.

1.8 Μοντελοποίηση πληροφορίας

Πριν ληφθεί υπόψη ο δεδομένος περιορισμός του Θέματος της παρούσης ΔΕ για χρήση OWL και SPARQL, αλλά και οι περιορισμοί που θα προκύψουν από τις επιλογές υλοποίησης, γίνεται μια αρχική ιεραρχία των υποψήφιων κλάσεων που προκύπτουν από τα επιλεγμένα μέχρι τώρα χαρακτηριστικά

1.8.1 Πιθανές Κλάσεις και Πληροφορίες

Προχωρώντας με βάση τα συστατικά της παραγράφου 1.5 προστίθενται τα απολύτως απαραίτητα στοιχεία δεδομένων και αντικειμένων κλάσεων σαν παράδειγμα χρήσης.

Μια **Εκδήλωση** έχει

- Ταυτότητα — Μοναδικό κλειδί
- Τίτλος — Δεδομένα
- Περιγραφή — Δεδομένα
- Ημερομηνία Έναρξης — Δεδομένα
- Ημερομηνία Λήξης — Δεδομένα
- Σκοπός — Αντικείμενο κλάσης Σκοπός
- Διοργανωτής — Αντικείμενο κλάσης Οντότητα
- Χώρο Place — Αντικείμενο κλάσης Χώρος
- Παράγωγο — Αντικείμενα κλάσης Παράγωγο(από 0 έως *)
- Συμβάν — Αντικείμενα κλάσης Συμβάν (από 1 έως *)

Μια **Οντότητα** έχει:

- Ταυτότητα — Μοναδικό Κλειδί
- Είδος Οντότητας — Αντικείμενο κλάσης Φυσικό Πρόσωπο
  - Όνομα — Δεδομένα
  - Επώνυμο — Δεδομένα

  ή

  Αντικείμενο κλάσης Νομικό Πρόσωπο
  - Τίτλο — Δεδομένα
  - Περιγραφή — Δεδομένα

Ένας **Σκοπός** έχει:

- Ταυτότητα — Μοναδικό κλειδί
- Τίτλο — Δεδομένα
- Περιγραφή — Δεδομένα

Ένας **Χώρος** έχει:

- Ταυτότητα — Μοναδικό κλειδί
- Περιγραφή — Δεδομένα

- Είδος Χώρου Αντικείμενο κλάσης «Φυσικός Χώρος»
  - Χώρα Δεδομένα
  - Πόλη Δεδομένα
  - Διεύθυνση Δεδομένα
  - Κτήριο Δεδομένα

ή

Αντικείμενο Κλάσης Εικονικός Χώρος
  - Πλατφόρμα Δεδομένα

Ένα **Παράγωγο** έχει:

- Ταυτότητα Μοναδικό κλειδί
- Τίτλο Δεδομένα
- Περιγραφή Δεδομένα

Ένα **Συμβάν** έχει:

- Ταυτότητα Μοναδικό κλειδί
- Τίτλος Δεδομένα
- Περιγραφή Δεδομένα
- Συμμετοχή Αντικείμενα κλάσης Συμμετοχή (1 έως *)

Μια **Συμμετοχή** έχει:

- Ταυτότητα Μοναδικό κλειδί
- Τίτλος Δεδομένα
- Περιγραφή Δεδομένα
- Συμμετέχοντα Αντικείμενο Κλάσης Οντότητα

1.8.2 Ειδίκευση, πληθυσμός και σύνθεση κλάσεων

Πραγματοποιήθηκαν δοκιμές με χαρτί και μολύβι σε ότι αφορά την ιεραρχία και το μέγεθος που πληθυσμού των αντικειμένων ανά κλάση.

Παρατηρήθηκε ότι κάποιες κλάσεις προβλέπεται να έχουν μεγαλύτερο αριθμό αντικειμένων από κάποιες άλλες. Συγκεκριμένα η κλάση «Συμμετοχή» θα έχει πολλές υλοποιήσεις ανά «Συμβάν» και θα υπάρχουν πολλά συμβάντα ανά εκδήλωση, με αποτέλεσμα να υπάρχουν λιμνάζοντα δεδομένα από υλοποιήσεις της κλάσης «Συμμετοχή» χωρίς να υπάρχει τρόπος ομαδοποίησης πέραν του «Συμβάντος» και της «Εκδήλωσης».

Αναθεωρήθηκε η σχεδίαση της κλάσης «Συμμετοχή» ως προς το να είναι μη υλοποιήσιμη υπερκλάση, κλάσεων συμμετοχής για να υπάρχει ομαδοποίηση στους τύπους συμμετοχής σε επίπεδο αντικειμένου.

Στην σχεδίαση προστέθηκαν οι υποκλάσεις «*Παράγοντας*», «*Γενική*» και «*Παρουσίαση*» και σημειώνεται ότι θα ήταν καλό να ορίζονται κλάσεις από τον χρήστη, οπού νομίζει ο χρήστης ότι χρειάζεται να γίνει.

Καταργείται το πεδίο «Συμμετοχή»

Μια <<**Συμμετοχή>>** δεν θα είναι υλοποιήσιμη (abstract) και θα έχει:

- Ταυτότητα — Μοναδικό κλειδί
- Τίτλος — Δεδομένα
- Περιγραφή — Δεδομένα
- Υποκλάση — Παράγοντας υλοποιήσιμο σε αντικείμενο
- Υποκλάση — Γενική υλοποιήσιμο σε αντικείμενο
- Υποκλάση — Παρουσίαση υλοποιήσιμο σε αντικείμενο
- Υποκλάσεις — Ορισμένες από την χρήστη

Δημιουργήθηκε σύνθεση κλάσεων «***Συμμετοχή Ως***», που αντιπροσωπεύει την πληροφορία «*Η Οντότητα 'Χ' συμμετέχει ως είδος Συμμετοχής 'Υ'*» που αντικαθιστά το πεδίο «*Συμμετοχή*» της κλάσης «Συμβάν».

Σημειώνεται ότι μία οντότητα, με αυτό τον τρόπο, μπορεί να συμμετέχει στο ίδιο συμβάν με παραπάνω από ένα είδος συμμετοχής π.χ. ο Χ συμμετέχει ως «*Γενική*» συμμετοχή αλλά και ως «*Παράγοντας*», αυτές είναι δύο διαφορετικές υλοποιήσεις (δύο διαφορετικά αντικείμενα)

Μια «**Συμμετοχή Ως**» έχει:

- Ταυτότητα — Μοναδικό κλειδί
- Τίτλος — Δεδομένα
- Είδος Συμμετοχής — Αντικείμενο κλάσης Συμμετοχή (υποκλάση της)
- Συμμετέχοντας — Αντικείμενο κλάσης Οντότητα

Αναθεωρήθηκε η σχεδίαση της κλάσης «*Συμβάν*», ένα «***Συμβάν*»** έχει:

- Ταυτότητα — Μοναδικό κλειδί
- Τίτλος — Δεδομένα
- Περιγραφή — Δεδομένα
- Συμμετοχή — Αντικείμενα κλάσης Συμμετοχή (1 έως *)
- Συμμετέχει — Αντικείμενο σύνθεσης «Συμμετοχή Ως» (1 έως *)

Παρατηρήθηκε από πειραματισμούς ότι είναι ωφέλιμη η περεταίρω εξειδίκευση της κλάσης «*Σκοπός*» για την πιο εύκολη ομαδοποίηση των εκδηλώσεων.

Αναθεωρήθηκε η σχεδίαση της κλάσης «*Σκοπός*» ως προς το να είναι μη υλοποιήσιμη υπερκλάση κλάσεων «*Σκοπού*», για να υπάρχει ομαδοποίηση στους τύπους σκοπού σε επίπεδο αντικειμένου. Στην σχεδίαση προστέθηκαν οι Υποκλάσεις «*Φιλανθρωπία*», «*Παρουσίαση Έκδοσης*», «*Επιχειρησιακή*» και σημειώνεται ότι θα ήταν καλό να ορίζονται κλάσεις από τον χρήστη.

Ένας <<**Σκοπός>>** δεν θα είναι υλοποιήσιμος (abstract) και θα έχει:

- Ταυτότητα Μοναδικό κλειδί
- Τίτλος Δεδομένα
- Περιγραφή Δεδομένα
- Υποκλάση Φιλανθρωπία υλοποιήσιμο σε αντικείμενο
- Υποκλάση Παρουσίαση υλοποιήσιμο σε αντικείμενο
- Υποκλάση Επιχειρησιακή υλοποιήσιμο σε αντικείμενο
- Υποκλάσεις Ορισμένες από τον χρήστη

1.9 Μοντελοποίηση UML

Για την αναπαράσταση της πληροφορίας χρησιμοποιήθηκε το **Unified Modeling Language (UML)[6],** που είναι μία γλώσσα μοντελοποίησης για την οπτική αναπαράσταση της σχεδίασης ενός συστήματος, σχεδιάστηκαν οι κλάσεις και οι σχέσεις τους που επιλέχθηκαν στο κεφάλαιο 1.8.1 και παρουσιάζονται ως ιδέα πριν την υλοποίηση του ΠΣ. Εξακολουθεί να μην θεωρείται ως περιορισμός (ή πλεονέκτημα) η χρήση οντολογιών, δεν προκύπτει κανένας περιορισμός ως προς την γλώσσα προγραμματισμού αλλά ούτε ως προς τον τρόπο αποθήκευσης και διαχείρισης των δεδομένων στον δίσκο.

1.9.1 Umbrello UML Modeler

Το Umbrello είναι ένα πρόγραμμα μοντελοποίησης που ακολουθεί τα στάνταρ της UML, είναι βασισμένο στην τεχνολογία της κοινότητας του **K Desktop Environment (KDE)[7]** και είναι δωρεάν λογισμικό ανοικτού κώδικα με υλοποιήσεις σε όλα τα δημοφιλή ΛΣ. Παρέχει ικανοποιητική **τεκμηρίωση[8]**, είναι εύκολο στην εγκατάσταση και στην χρήση.

Η κοινότητα χρηστών δημιούργησε και υποστηρίζει φορητή έκδοση (έκδοση που δεν χρειάζεται εγκατάσταση), η οποία χρησιμοποιήθηκε σε αυτήν την ΔΕ, με ημερομηνία έκδοσης φορητού προγράμματος 27 Αυγούστου 2020.

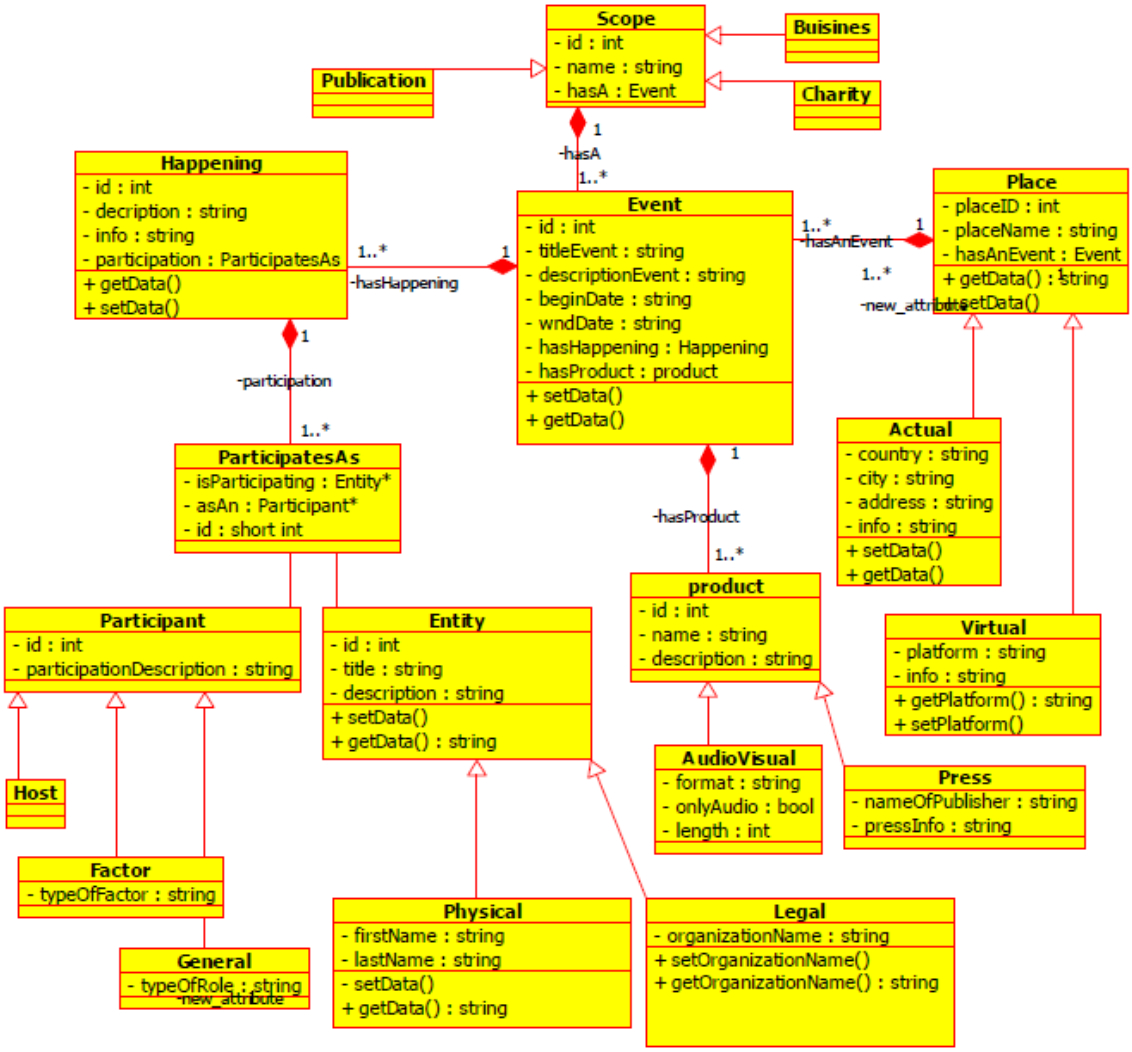
1.9.2 Γραφική αναπαράσταση κλάσεων και σχέσεων

Παρουσιάζεται *στο Σχήμα 1.1* μια αρχική υλοποίηση σε UML της σχεδίασης της παραγράφου 1.8, γίνεται παρουσίαση των σχέσεων μεταξύ των κλάσεων και των εξαρτήσεων τους καθώς και παράσταση των πληθικών εξαρτήσεων.

Έγινε χρήση λατινικών χαρακτήρων και τα πεδία δεδομένων είναι ενδεικτικά, προστέθηκαν δοκιμαστικά υποκλάσεις της κλάσης «Παράγωγο» (product) : Audio Visual και Press.

Θεωρήθηκε σαν μοναδικό μη Σημασιολογικά υλοποιήσιμο χαρακτηριστικό που σχήματος η ύπαρξη του πεδίου ID, που προσφέρει μεν την μοναδικότητα σε ένα κλασσικό ΠΣ αλλά, δεν προσφέρει καμία λειτουργικότητα που να συμπλέει με την λογική της Οντολογικής αναπαράστασης.

Θα θεωρηθεί στην συνέχεια ως **αναγνωριστικό μοναδικότητας το URI**, και όχι το πεδίο ID



Σχήμα 1.1: Αρχική υλοποίηση UML

1.10 Συμπεράσματα πριν την υλοποίηση

Η όλη ανάλυση του Κεφαλαίου 1 έγινε με τον πιο γενικευμένο τρόπο που θα μπορούσε, ώστε να είναι υλοποιήσιμο το ΠΣ διαχείρισης εκδηλώσεων σε όλα τα γνωστά (από τον συντάκτη της Διπλωματικής) Framework και γλώσσες προγραμματισμού.

Η συνέχεια είναι η επιλογή του τρόπου αποθήκευσης των δεδομένων, η επιλογή του περιβάλλοντος υλοποίησης και η σχεδίαση διεπαφών χρήστη.

Κεφάλαιο 2ο: Τεχνολογίες υλοποίησης

Μελετάται σε αυτό το κεφάλαιο το τεχνολογικό πλαίσιο του ΣΙ γενικά, παρουσιάζονται οι βασικές δομές δεδομένων, τα συστατικά μιας οντολογίας, οι γλώσσες υλοποίησης, οι βιβλιοθήκες αλλά και τα βοηθητικά εργαλεία ανάπτυξης.

2.1 Τεχνολογικές προϋποθέσεις

Η προσέγγιση της υλοποίησης έχει παρονομαστή τον Σημασιολογικό Ιστό, γεγονός που φιλτράρει τις διαθέσιμες τεχνολογίες. Αυτές οι τεχνολογίες θα πρέπει να έχουν Σημασιολογική συμβατότητα ή απλά να μην είναι αποτρεπτικές στην χρήση οντολογιών. Ερευνήθηκαν οι γλώσσες προγραμματισμού και η ύπαρξη βιβλιοθηκών του ΣΙ.

Σύμφωνα με το W3C, οι παρακάτω γλώσσες προγραμματισμού επιτρέπουν την ανάπτυξη ή την **χρήση εργαλείων του Σημασιολογικό Ιστού [9]:**

ActionScript	OpenLink Virtuoso
C-sharp	Redland RDF application Framework
Lisp	AllegroGraph Store
Prolog	AllegroGraph Store
Obj-C	1- OpenLink Virtuoso 2- Redland RDF application Framework
Tcl	1- OpenLink Virtuoso 2- Redland RDF application Framework
Javascript	1- Dojo.data2 2- Mobi 3- OpenLink Virtuoso
Ruby	1- OpenLink Virtuoso 2- Redland RDF application Framework 3- AllegroGraph RDF Store
C	1- Redland RDF application Framework 2- GraphDB 3- OpenLink Virtuoso 4- AllegroGraph RDF Store
PHP`	1- Redland RDF application Framework 2- OpenLink Virtuoso 3- Sesame 4- Outdated-ARC RDF Store
Python	1- AllegroGraph RDF Store 2- OpenLink Virtuoso 3- RDFLib 4- Redland RDF application Framework 5- Sesame
Java	1- AllegroGraph RDF Store

2- Apache Jena
3- Mobi
4- Mulgara Semantic Store
5- OpenLink Virtuoso
6- Oracle Spatial and Graph 19c
7- GraphDB
8- RDFox
9- Redland RDF application Framework
10-Sesame

Παρατηρείται η πληθώρα εργαλείων για την Java, αλλά και η παρουσία του **OpenLink Virtuoso** σχεδόν σε όλες τις γλώσσες. Ελάχιστες επιλογές δεν καλύπτονται από την Java όπως το RDFLib για Python, Outdated-ARC rdf Store για PHP και Dojo.data για Javascript, τα οποία είναι εργαλεία προσανατολισμένα προς τις γλώσσες τους και οι λειτουργίες τους καλύπτονται από την βιβλιοθήκη **Jena του Apache Foundation[10].**

2.2 Η γλώσσα προγραμματισμού Java

Είναι από τις **πιο γνωστές και δημοφιλείς γλώσσες προγραμματισμού[11]**, σύμφωνα με το GitHub από το 2019 και μετά, είναι η πιο χρησιμοποιούμενη γλώσσα στην πλατφόρμα του. Οι απαιτήσεις που έχει είναι η ύπαρξη του Java Virtual Machine περιβάλλοντος για να εκτελέσει ένα πρόγραμμα, ρίχνοντας το βάρος της συμβατότητας στο περιβάλλον και όχι στον προγραμματιστή που γράφει Java.

Ουσιαστικά το JVM περιβάλλον εκτελεί κώδικα byte code, ο κώδικας αυτός προέρχεται από το compile του κώδικα java. Ο ισχυρισμός των δημιουργών της Java είναι (WORA) **"το γράφεις μια φορά και τρέχει παντού"[12]** και θα θεωρηθεί από τον συντάκτη αυτού του κειμένου ως αληθής έχοντας εκτελέσει χωρίς πρόβλημα, προγράμματα Java σε διαφορετικά Λ/Σ και υπολογιστές με διαφορετική αρχιτεκτονική, με έναν μικρό αστερίσκο σε σχέση με τις μεταβλητές περιβάλλοντος και την πρόσβαση σε αυτές.

Οποιαδήποτε αμφιβολία για την επιτυχία της ιδέας του WORA μέσω JVM δείχνει να χάνεται όταν διαπιστώνεται το 2001 ότι **γλώσσες όπως[13]**: Groovy, Scala, Kotlin, Closure παράγουν byte code ο οποίος εκτελείται από το JVM, αλλά και όταν εμφανίζονται για παλαιότερες και νέες γλώσσες προγραμματισμού **υλοποιήσεις σε JVM[14]** (Java ουσιαστικά) όπως:

- Jpython για Python
- Jabaco για Visual Basic
- Rhino, Nashrom και Graal.js για JavaScript
- Quercus JPHP για PHP
- Renjin για R
- JRuby για Ruby
- Neo4j για Cypher
- JIProlog και TuProlog για Prolog
- MIDIetPascal Oxygene για Pascal
- Micro Focus Visual COBOL, Heirloom Elastic και Veryant isCobol Evolved για COBOL

Η **οντολογική παρουσία** της Java είναι θεμελιώδης, ένα από τα πιο δημοφιλή και εύχρηστα γραφικά περιβάλλοντα ορισμού και ελέγχου Οντολογιών, το **Protégé[15]**, πέραν του ότι είναι γραμμένο σε Java και τρέχει σε JVM, δίνει στον δημιουργό της οντολογίας την δυνατότητα να εξάγει **την δομή μιας Οντολογίας** σε σωστά διαμορφωμένες δηλώσεις **Java**.

Πλήρες framework για την ανάπτυξη και διαχείριση Σημασιολογίας παρέχεται στην Java από το **Apache Jena**.

Ο πακτωλός βιβλιοθηκών, εργαλείων, κοινοτήτων και γενικά πληροφορίας που υπάρχει σε σχέση με την Java αλλά και η εμπλοκή του **Apache Software Foundation** αποτέλεσε τον καθοριστικό παράγοντα στην επιλογή της πλατφόρμας για την υλοποίηση αυτής της Εργασίας υπό το πρίσμα της Οντολογικής διαχείρισης.

2.3 Δομές δεδομένων και πληροφορίας

Η σημερινή αποτύπωση μιας πληροφορίας με Οντολογική Σημασία βασίζεται σε υπάρχουσες δομές που εφαρμόζονται άμεσα ή κληρονομήθηκαν από τις νέες θέτοντας ουσιαστικά το αλφαβητάρι της εξέλιξης της.

2.3.1 Extensible Markup Language

Το W3C το 1998 προτυποποίησε την **XML1.0 [16]** και το 2008 προσδιόρισε την πέμπτη έκδοση της **XML1.0 [17]** με σκοπό την καλύτερη λειτουργικότητα τόσο με την HTML όσο και με την SGML.

Η XML1.1 ξεκίνησε το 2004 και βρίσκεται από το 2006 στην **δεύτερη έκδοση[18]** της αλλά δεν είναι διαδεδομένη η χρήση της παρά μόνο αν υπάρχει ανάγκη χρήσης ειδικών χαρακτήρων και script.

Είναι μια markup γλώσσα που ορίζει ένα σύνολο από κανόνες συγγραφής εγγράφων με μια μορφή που να είναι αναγνώσιμη από ανθρώπους αλλά και από μηχανές, σαν μηχανή νοείται το λογισμικό υπολογιστών που θα πάρει σαν είσοδο το εν λόγω έγγραφο. Μηχανές και Άνθρωποι μπορούν να ερμηνεύσουν το έγγραφο βάσει αυτού του συνόλου κανόνων ασχέτως με το πλαίσιο ανάπτυξης, την γλώσσα ή το περιβάλλον εφαρμογής.

2.3.1.1 XML δομή

Αν και δομικά μοιάζει με την HTML, αφού μοιράζεται την χρήση των tag, τα tag αυτά δεν είναι προκαθορισμένα όπως στην HTML αλλά είναι ελεύθερα (από σημασία) και οριζόμενα από τον συντάκτη του εγγράφου. Η σημασία των tag αφορά τον ορισμό του ίδιου εγγράφου και των σχέσεων που προκύπτουν, ενώ στην HTML τα tag ορίζουν την μορφοποίηση της εμφάνισης του υπό διερμήνευση εγγράφου από τον browser.

Σκοπός της είναι η απλότητα και η γενικοποίηση για την χρήση της στον Παγκόσμιο Ιστό, έχει μορφή απλού κειμένου και θα μπορούσε να αποτυπωθεί σε δενδροειδή δομή, αναπαριστά εύκολα δεδομένα με όχι τόσο αυστηρές δομές, ακόμα και αυθαίρετες, συμπεριλαμβάνει ισχυρή υποστήριξη Unicode για χρήση της από διάφορες Εθνικές γλώσσες.

Το Tag ορίζεται να αρχίζει με “<” και τελειώνει με “>” συγκεκριμένα υπάρχουν τα tag

Έναρξης π.χ. **<human>**

και τέλους π.χ. **</human>**

Οι γραμμές κειμένου ανάμεσα σε αυτά τα δύο tag περιγράφουν ένα μέρος του εγγράφου που θεωρείται στοιχείο (element) του εγγράφου, φυσικά η σημασία του tag και του element περιορίζεται στην συμφωνία που υπάρχει μεταξύ συντάκτη και αναγνώστη.

Ορίζεται και κείμενο ανάμεσα σε “&” και ένα “;”

Το “***A***” είναι το “*A*”

Το "***<***" είναι το "<"

Το "***&amp***" είναι το "&"

κοκ, χωρίς την ειδική σημασία που έχουν σε ένα έγγραφο XML, δηλαδή γίνεται αναπαράσταση των χαρακτήρων ως χαρακτήρες και μόνο.

Υπάρχουν tag που δεν είναι αρχής ή τέλους αλλά περιέχουν την πληροφορία που χρειάζεται ανάμεσα στους χαρακτήρες "<" και "/>" **<alien name="Alf" />**

Τα σχόλια ορίζονται σαν **<!--αυτό είναι ένα σχόλιο -->**

Η κεφαλίδα ορίζεται σαν:

<?xml version="1.0"?> ή

<?xml version="1.0" encoding="UTF-16"?>

2.3.1.2 Document Type Definition

Είναι ένα σύνολο από δηλώσεις που ορίζουν τον τύπο του XML(GML,SGML,HTML) εγγράφου. Ορίζοντας ουσιαστικά τα δομικά στοιχεία του εγγράφου μέσω μιας λίστας από επικυρωμένα και προσυμφωνημένα στοιχεία και χαρακτηριστικά. Μπορεί να είναι μέρος του XML εγγράφου αλλά μπορεί να είναι **εξωτερικό έγγραφο[19]** ώστε να γίνεται χρήση από πολλά XLM έγγραφα.

Η δήλωση γίνεται στην αρχή του XML εγγράφου με το αναγνωριστικό DOCTYPE και κάνει γνωστό (σε Άνθρωπο και Μηχανή) ότι το XML έγγραφο είναι μια υλοποίηση των τύπων του αναφερόμενου DTD.

Παρακάτω ορίζεται ένα XML 1.0 έγγραφο με encoding UTF-8 στο οποίο υπάρχει εξάρτηση από εξωτερικό αρχείο DTD, το αρχείο αυτό ορίζει τον τρόπο δήλωσης και τα χαρακτηριστικά ενός στοιχείου <event_list> που εν' δυνάμει μπορεί να δηλωθούν στο κυρίως σώμα του XML.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE ebent_list SYSTEM "event.dtd">
<event_list />
```

Ενδεικτικά παρατίθεται μέρος της μορφής του αρχείου "event.dtd":

```
<!ELEMENT event_list (event)*>
    <!ELEMENT event (name, description?, startDate?, endDate?, scope? )>
        <!ELEMENT name (#PCDATA)>
        <!ELEMENT description (#PCDATA)>
        <!ELEMENT startDate (#PCDATA)>
        <!ELEMENT endDate (#PCDATA)>
        <!ELEMENT scope (#PCDATA)>
```

Η υλοποίηση στο κυρίως κείμενο του XLM θα μπορούσε ενδεικτικά να είναι:

```
<event_list>
    <event>
        <name> Τα βιβλία του Philip Kindred Dick</name>
        <description>
            Η συμβολή στον φουτουριστικό στοχασμό του Philip K. Dick
        </description>
        <startDate> 2021-07-01 </startDate>
        <endDate> 2021-07-01 </endDate>
        <scope>Εναλλακτική Έρευνα </scope>
    </event>
    <event>
        <name> The Simulation Hypothesis</name>
        <description>
            Διάλεξη του N. Bostrom για την επιβίωση ενός πολιτισμού
            μέχρι την εξέλιξη του σε μέτα-ανθρώπινο  πολιτισμό και η
            πιθανότητα δημιουργίας προγονικής εξομοίωσης από αυτόν.
        </description>
        <startDate> 2021-07-31 </startDate>
        <endDate> 2021-08-01 </endDate>
        <scope> Ομιλία  </scope>
    </event>
</event_list>
```

Ορίζει μια λίστα εκδηλώσεων, ή καλύτερα μια λίστα <event_list>,με πληροφορίες που έχουν επιλεχθεί για την εκδήλωση σχετικά με τα βιβλία του P.K.Dick και την εκδήλωση (ομιλία) του καθηγητή N. Bostrom. Πλέον Άνθρωπος και Μηχανή γνωρίζοντας τις δηλώσεις στο DTD περιμένουν να βρούνε στοιχεία στο XML με την σημασία του έχει ορισθεί σε αυτά. Γνωρίζει ότι υπάρχει μια λίστα από στοιχεία, γνωρίζει τα "μνημονικά" τους χαρακτηριστικά και μπορεί να εξάγει στοιχειοθετημένη πληροφορία από το XML έγγραφο.

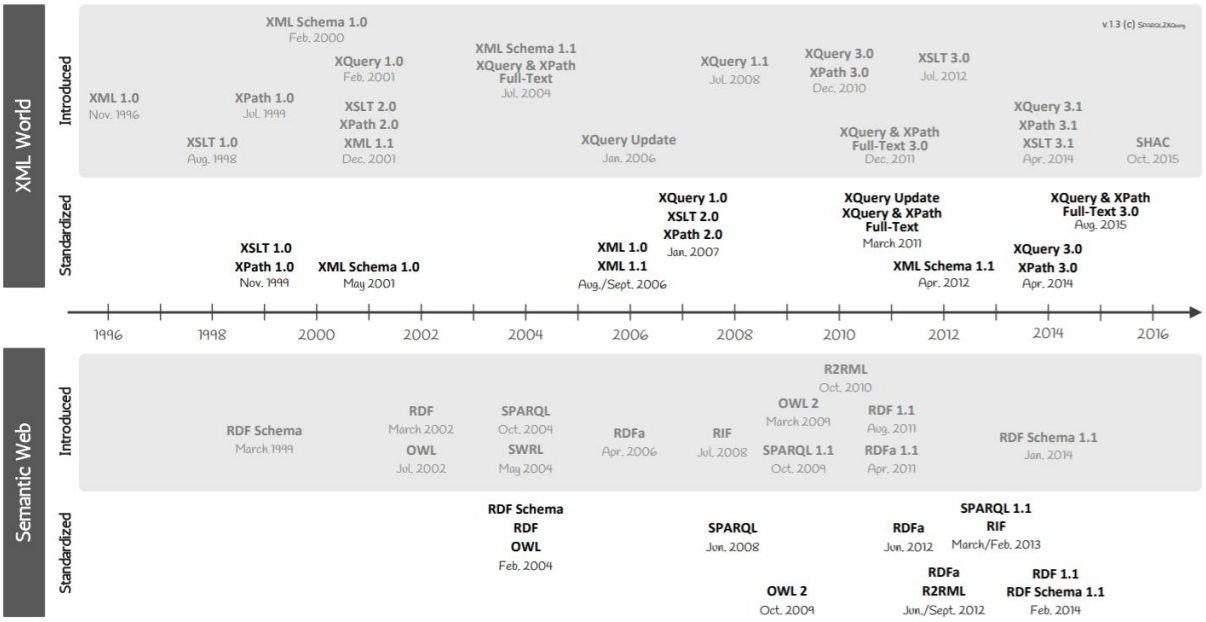
2.3.1.3 Η μετά XML εποχή

Η πορεία των XML δομών δεδομένων προηγείται από τον Σημασιολογικό Ιστό, επεκτείνεται σε πολλά επίπεδα και έχει την δική της εξέλιξη μέχρι και σήμερα. Πρόκειται για δύο διαφορετικούς αλλά παράλληλους κόσμους που ο διαχωρισμός τους έγινε με την εμφάνιση του Resource Description Framework.

Μετά το XML υπήρξαν το XQuery που είναι μια γλώσσα ερωτημάτων με functional χαρακτηριστικά, το XSLT που επιτρέπει την μετατροπή των XML εγγράφων σε άλλες μορφές όπως HTTP, xls και

άλλα. Το XPath 2.0 παρουσιάστηκε το 2007, ως γλώσσα ερωτημάτων με σκοπό την επιλογή κόμβων από xml έγγραφο το XML Scheme 1.1 το 2012, η 3.0 έκδοση των XQuery & XPath 3.0 το 2014.

Η **παράλληλη εξέλιξη[20]** αποτυπώνεται στο *Σχήμα 3.1*



Σχήμα 3.1: XML and Semantic Web W3C Standards Timeline-History

2.3.2 Resource Description Framework

Στον πραγματικό κόσμο, που αντιλαμβάνεται ένα ανθρώπινο ον, ένα αντικείμενο και μια περιγραφή δεν θα μπορούσαν να είναι σχετικά χωρίς την ύπαρξη της σύνδεσης τους.

Μια μητέρα δείχνει στο παιδί της το οπτικό ερέθισμα ενός ελέφαντα και του δίδει το ακουστικό ερέθισμα "*Ελέφαντας*". Δημιουργείται μια σύνδεση γνώσης, που αποδίδει πληροφορία και χαρακτηριστικά σε αντικείμενο. Αυτή όμως η περίπτωση βάζει σε λειτουργία πολύπλοκους μηχανισμούς αισθήσεων (όραση, ακοή) και αισθημάτων (εμπιστοσύνη) αλλά δεν παύει να είναι επί της ουσίας μια "απλή" **δήλωση**.

Με αφηρημένο τρόπο ορίζεται μια τριάδα :

Υποκείμενο Κατηγόρημα Αντικείμενο

Subject Predicate Object

Το Κατηγόρημα **ορίζει** μια ιδιότητα του Υποκειμένου, ιδιότητα αυτή **είναι** το Αντικείμενο.

Η εικόνα του ελέφαντα (ο ελέφαντας) είναι το Υποκείμενο, το Κατηγόρημα είναι το νεύμα της μητέρας που δείχνει το Υποκείμενο και υπονοεί "*αυτό το λέμε* ", το Αντικείμενο είναι ο ήχος που παράγεται από την μητέρα λέγοντας την λέξη "*Ελέφαντας*", καταλήγουμε στην τριάδα:

"Αυτό", "το λέμε", "Ελέφαντα"

Η περιγραφή πόρων είναι ένα πλαίσιο που έχει διάφορες μορφές σύνταξης, στο RDF υπάρχει αναπαράσταση από **XML στοιχείο** που έχει ετικέτα **rdf:RDF** και αποδίδει έναν Γράφο.

<?xml version="1.0"?>

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

xmlns:base="http://allios.co.nf/">

<rdf:Description rdf:about="http://www.myanimals.org/elephant/">

<base:titleEn>Elephant</base:titleEn>

<base:titleEl>Ελέφαντας</base:titleEl>

</rdf:Description>

</rdf:RDF>

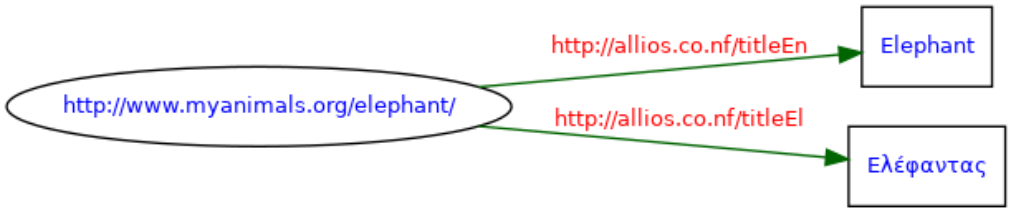
Ο πόρος είναι το http://www.myanimals/elephant/, οι ιδιότητες είναι ο τίτλος στα αγγλικά (titleEn) και ο τίτλος στα ελληνικά (titleEl).

Μέσω του **https://www.w3.org/RDF/Validator/** δοκιμάζεται η παραπάνω δήλωση με το παρακάτω αποτέλεσμα στον *Πίνακα 2.1* και στο *Σχήμα 2.2*.

Πίνακας 2.1: Triplets of the Data Model

Number	Subject	Predicate	Object
1	http://www.myanimals.org/elephant/	http://allios.co.nf/titleEn	"Elephant"
2	http://www.myanimals.org/elephant/	http://allios.co.nf/titleEl	"Ελέφαντας"

Graph of the data model



Σχήμα 2.2: Validation από το www.w3.org

2.3.3 RDF Schema

Η πρώτη έκδοση του **RDFS[21]** δημοσιεύτηκε από το W3C το 1998 και η τελευταία το **2014[22]**.

Είναι ένα σύνολο από κλάσεις και ιδιότητες υπό το μοντέλο δεδομένων RDF που παρέχει τα βασικά στοιχεία για την περιγραφή μιας οντολογίας, από εκεί εξάγονται πληροφορίες χρησιμοποιώντας γλώσσα ερωτημάτων όπως η SPARQL.

Τα βασικά στοιχεία του RDF Schema :

Classes (Κλάσεις)

- rdfs:Resource
  - Ότι περιγράφεται από RDF είναι πόρος.
- rdfs:Class
  - Ένας πόρος τύπου rdfs:Class είναι ένας πόρος που μπορεί να χρησιμοποιηθεί σαν Κλάση για άλλους πόρους
- rdfs:Literal
  - Τιμές αριθμητικές, αλφαριθμητικές, και σταθερές και όχι εξαρτώμενες από μεταβλητές.
- rdfs:Datatype
  - Είναι πόροι που είτε είναι τύπου rdfs:Class είτε προκαθορισμένου τύπου rdfs:Literal

- rdf:XMLLiteral
  - Είναι πόροι που είναι τύπου rdfs:Datatype
- rdf:Property
  - Η κλάση των ιδιοτήτων

Properties - Ιδιότητες

Είναι υλοποιήσεις της κλάσης rdf:Property και περιγράφει σχέσεις μεταξύ Υποκειμένων και Αντικειμένων.

- rdfs:domain
  - Πόρος τύπου rdf:Property που ορίζει την Κλάση που μπορεί να ανήκει το Υποκείμενο σε μια τριάδα
- rdfs:range
  - Πόρος τύπου rdf:Property που ορίζει την Κλάση που μπορεί να ανήκει το Αντικείμενο σε μια τριάδα
- rdf:type
  - Ιδιότητα που ορίζει ότι ένας πόρος είναι μια υλοποίηση μιας Κλάσης
- rdfs:subClassOf
  - Ιδιότητα που δείχνει την ιεραρχία μιας κλάσης
- rdfs:subPropertyOf
  - Ιδιότητα που δείχνει την ιεραρχία των ιδιοτήτων
- rdfs:label
  - Ιδιότητα που δίνει πληροφορίες για τον πόρο
- rdfs:comment
  - Ιδιότητα που δίνει σχόλια για τον πόρο

Utility properties

- rdfs:seeAlso
  - Παραπομπή για πληροφόρηση σε άλλον πόρο
- rdfs:isDefinedBy
  - Παραπομπή σε εξωτερικό λεξικό ή σε πληροφορία του πόρου

2.4 Σημασιολογικός Ιστός

Το πρώτο επίσημο άρθρο που περιγράφει τον Σημασιολογικό Ιστό δημοσιεύτηκε το 2001 στο περιοδικό **Scientific American[23]**, περιγράφοντας πως η επέκταση του διαδικτύου θα έδινε πολύ περισσότερη δύναμη στα δεδομένα. Για να επιτύχουμε αυτήν τη επέκταση θα πρέπει να δοθεί τέτοια δομή στο νοηματικό περιεχόμενο των ιστοσελίδων, ώστε να δημιουργήσουμε έτσι ένα περιβάλλον στο οποίο οι μηχανές που περιπλανώνται στις σελίδες να μπορούν να εκτελέσουν πιο εκλεπτυσμένες εργασίες για τους χρήστες. Αυτή τη στιγμή, την προσπάθεια υλοποίησης του Σημασιολογικού Ιστού ηγείται η Παγκόσμια Διαδικτυακή Κοινοπραξία **W3C[24]** (World Wide Web Consortium). Στην ιστοσελίδα της ως άνω αναφερόμενης κοινοπραξίας δίνεται η εξής εξήγηση για τον Σημασιολογικό Ιστό:

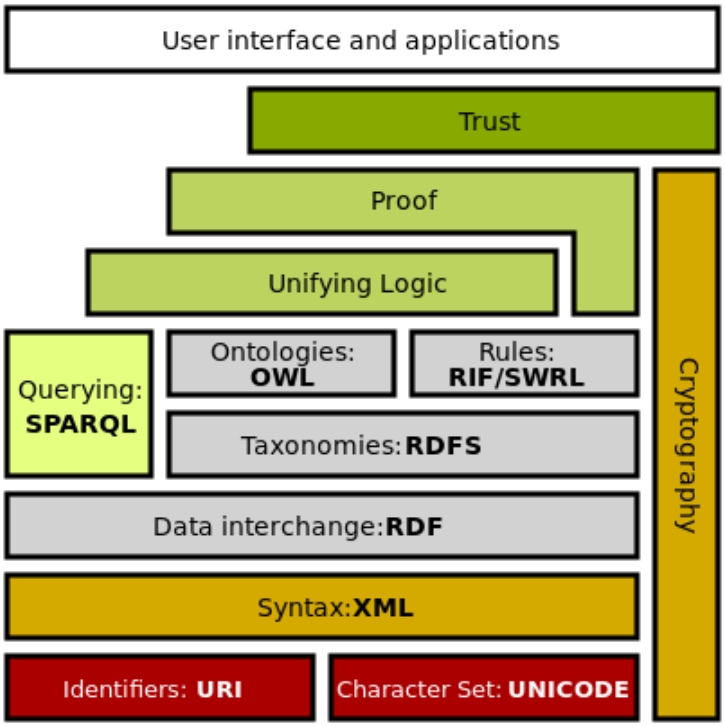
Ο Σημασιολογικός Ιστός παρέχει ένα κοινό πλαίσιο εργασίας το οποίο επιτρέπει τον διαμοιρασμό δεδομένων και την επανάχρησή τους σε εφαρμογές, επιχειρήσεις και κοινοτικά όρια.

Σε γενικές γραμμές αυτό σημαίνει ότι τα δεδομένα στο διαδίκτυο θα πρέπει να είναι κατανοητά στους ανθρώπους αλλά και στις μηχανές. Εντούτοις, επίσημος ορισμός του Σημασιολογικού Ιστού δεν

υπάρχει. Όπως αναφέρει ο **Thomas B. Passin[25]** στο βιβλίο του «Explorer's Guide to the Semantic Web", ο Σημασιολογικός Ιστός δεν αποτελεί τεχνολογία αλλά όραμα. Οι B. **Hoenderboom και P. Lang[26]** μας εξηγούν ότι ο ΣΙ είναι ένας ιστός οντοτήτων, και όχι ένας ιστός εγγράφων, όπως συμβαίνει με τον τρέχοντα ιστό. Οι οντότητες και οι σχέσεις τους είναι διαθέσιμες σε ολόκληρο το Διαδίκτυο. Για να υλοποιήσουν έναν σημασιολογικό ιστό, οι σημασιολογίες προστίθενται στον υπάρχοντα ιστό μέσω οντολογιών που φέρουν μεταδεδομένα. Με τη χρήση της σημασιολογίας είναι δυνατό να συσχετιστούν αυτόματα μεταξύ τους διαφορετικές οντότητες. Όπως αναφέρθηκε ήδη, αυτές οι οντότητες μπορούν να είναι διαθέσιμες σε ολόκληρο το Διαδίκτυο και δεν περιορίζεται σε μια ενιαία εφαρμογή ή ιστοσελίδα.

2.4.1 Αρχιτεκτονική

Την βασική αρχιτεκτονική του Σημασιολογικού Ιστού αποτελεί η **Στοίβα ΣΙ (Semantic Web Stack) [27]**, η οποία αποτυπώνεται στο *Σχήμα 2.3*



Σχήμα 2.3: The Semantic Web Stack

Στην βάση υπάρχουν οι **Identifiers** URI που ορίζουν έναν πόρο και το Character **Set,** την **σύνταξη** την αναλαμβάνει η XML και την **διαχείριση** των δεδομένων το RDF, αναλύθηκαν στις παραγράφους 2.3.1 και 2.3.2 αντίστοιχα. Στο επόμενο επίπεδο, όπως αναφέρεται στην παράγραφο 2.3.3, έχουμε την εφαρμογή του RDFS και μαζί εμφανίζονται οι κανόνες και οι Οντολογίες. Η πληροφορία σε αυτό το επίπεδο είναι προσβάσιμη μέσω γλώσσας ερωτημάτων, την SPARQL, που αναλύεται παρακάτω και θα γίνει χρήση της στο Π.Σ. Διαχείρισης Εκδηλώσεων με Οντολογίες.

Μέχρι το επίπεδο της διεπαφής χρήστη και εφαρμογών υπάρχουν τα επίπεδα **Unifying Logic, Proof και Trust:** Η λογική (Logic) συλλογιστική χρησιμοποιείται για να διαπιστωθεί η συνοχή και η ορθότητα των δεδομένων. Οι αποδείξεις (Proof) εντοπίζουν ή εξηγούν τα βήματα λογικής συλλογιστικής ενώ η εμπιστοσύνη (Trust) είναι ο τρόπος για την παροχή της γνησιότητας και των αποδεικτικών στοιχείων της αξιοπιστίας των δεδομένων.

2.4.2 Το μέλλον του Σ.Ι.

Η πλήρης εκμετάλλευση όμως των δυνατοτήτων που προσφέρει ο Σημασιολογικός Ιστός προϋποθέτει την ολοκλήρωση τουλάχιστον τριών βασικών βημάτων, όπως αυτά απεικονίζονται στο σχήμα 3.2. Τη δόμηση της πληροφορίας με στόχο την αναπαράστασή της ως γνώση, την επεξεργασία της με στόχο την εξαγωγή λογικών αποτελεσμάτων, της απόδειξής τους και τέλος την εδραίωση της εμπιστοσύνης ως προς την εγκυρότητα της ανακτηθείσας πληροφορίας **[28]**.

Η πραγματική δύναμη του Σημασιολογικού Ιστού γίνεται αντιληπτή με την ανάπτυξη προγραμμάτων λογισμικού, τα οποία ονομάζονται πράκτορες (agents). Αυτά τα προγράμματα συλλέγουν περιεχόμενα από διάφορες πηγές, τα επεξεργάζονται και ανταλλάσσουν τα αποτελέσματα με άλλα προγράμματα. Η αποτελεσματικότητα αυτών των πρακτόρων θα αυξάνεται εκθετικά καθώς ολοένα και περισσότερα από τα περιεχόμενα του Παγκόσμιου Ιστού γίνονται κατανοητά από τις μηχανές.

2.5 Οντολογία

Στην επιστήμη των υπολογιστών αλλά και στην φιλοσοφία η οντολογία έχει σαν βασική λειτουργία, την προσπάθεια αναπαράστασης Οντοτήτων, Ιδεών και Γεγονότων με όλες τις ιδιότητες τους, τις εξαρτήσεις τους αλλά και τον συσχετισμό αυτών, πάντα με ένα σύστημα κατηγοριοποίησης και διακριτούς κανόνες.

Σύμφωνα με τον **Gruber[29]** :

> *"An ontology is a description (like a formal specification of a program) of the concepts and relationships that can formally exist for an agent or a community of agents. This definition is consistent with the usage of ontology as set of concept definitions, but more general. And it is a different sense of the word than its use in philosophy. "*

Βάσει της παραπάνω άποψης, ένας πράκτορας (agent), όπως αναφέρθηκε στην παράγραφο 2.4.2, ή μια κοινότητα τους εκτίθεται στην ύπαρξη της τυπικής περιγραφής μιας οντολογίας σαν ένα σύνολο γενικευμένων εννοιών με τυπικούς κανόνες σχέσεων και ιδιοτήτων, κάτι διαφορετικό με την φιλοσοφική προσέγγιση.

2.5.1 Συστατικά της οντολογίας

Τα συστατικά μιας οντολογίας, πέρα από πληροφορία περιέχουν και περιγράφουν σχέσεις, περιορισμούς, κανόνες, ιδιότητες.

Individuals

Είναι υλοποιήσεις αντικειμένων**,** δεν θα γίνει η χρήση του όρου «**Άτομο**» κατ' επιλογή για την αποφυγή σύγχυσης των εννοιών της Οντολογίας Εκδηλώσεων π.χ. Individual είναι ο «*Γιάννης Παπαδόπουλος*» αλλά Individual είναι και το «*Συγκέντρωση Χρημάτων για αστέγους*», το μεν είναι υλοποίηση κλάσης "**Entity**" το δε είναι υλοποίηση κλάσης "**Scope**".

Classes

Είναι τα «*Είδη*», «*Η κατηγορία*» των αντικειμένων βάσει των οποίων υλοποιούνται τα Individuals, θα χρησιμοποιηθεί ο όρος «**Κλάση**» με την οπτική του «*Κλάση Αντικειμένου*».

Attributes

Είναι τα «*Χαρακτηριστικά*», οι «*Πτυχές*» οι «*Παράμετροι*» που μπορεί να έχει η κλάση ενός Individual, θα γίνει η χρήση του όρου «**Ιδιότητες**».

Relations

Ο τρόπος που σχετίζονται κλάσεις και Individuals μεταξύ τους, επιλέχθηκε ο Όρος «**Σχέση**».

Function terms

Σύνθεση Σχέσεων που μπορούν να χρησιμοποιηθούν σαν μια απλή Σχέση, επιλέχθηκε να χρησιμοποιηθεί ο όρος ως έχει.

Restrictions

Δηλώσεις που γίνονται αρχικά και πρέπει να ικανοποιούνται ως αληθής, για να ισχύει σαν αποδεκτό, κάποιο στοιχείο στην οντολογία, επιλέχτηκε ο Όρος «**Περιορισμοί**».

Rules

Δηλώσεις της μορφής «*Αν ισχύει ... τότε ...*» που περιγράφει ένα λογικό συμπέρασμα ενός ισχυρισμού, επιλέχθηκε ο όρος «**Κανόνες**».

Events

Είναι η αλλαγή σχέσεων ή ιδιοτήτων, επιλέχθηκε ο Όρος «**Τροποποίηση**».

Axioms

Είναι λογικοί ισχυρισμοί που επιβεβαιώνονται ή επιβεβαιώνουν την δομή όλης ή μέρους της Οντολογίας, θα μπορούσε να πει κάποιος ότι μια οντολογία είναι ένα σύνολο αξιωμάτων, αλλά και ότι τα αξιώματα επιβάλλουν ισχυρισμούς επί της λογικής της Οντολογίας, μερικές φορές ενάντια στον ισχυρισμό ενός λογικού συμπεράσματος (reasoning), που προκύπτει από άλλες δηλώσεις και κανόνες της Οντολογίας.

Τα αξιώματα χωρίζονται σε κατηγορίες και παρατίθενται μερικά σαν παράδειγμα :

Class

SubClassOf, EquivalentClasses, DisjointClasses, GCI count

Individual

ClassAssertion, ObjectPropertyAssertion, DataPropertyAssertion, NegativeObjectPropertyAssertion, NegativeDataPropertyAssertion, SameIndividual, DifferentIndividuals.

Data Property

DataPropertyRange, DataPropertyDomain, SubDataProperty, FunctionalDataProperty, DisjointDataProperties, EquivalentDataProperties.

Object Property

ObjectPropertyRange, ObjectPropertyDomain, FunctionalObjectProperty, InverceObjectProperty, InverceFunctionalObjectProperty, SymmetricObjectProperty, AsymmetricObjectProperty, EquivalentObjectPropery.

Annotation

AnnotationAssertion, AnnotationPropertyDomain, AnnotationPropertyRangeOf.

Έγινε κατανοητό από πειραματισμό, και τίθεται ως παράδειγμα που περιγράφει ένα αξίωμα, ότι: Μπορεί οι δηλώσεις που έχουν γίνει σε μια οντολογία να οδηγούν στο συμπέρασμα ότι το Individual 1 βάσει των χαρακτηριστικών του είναι (το) ίδιο με το Individual 2, αλλά με το αξίωμα DiferentIndividuals αποτρέπεται να γίνει αυτός ο συμπερασμός. Με τον ίδιο τρόπο όμως μπορεί να προκύπτει το αξίωμα DiferentIndividuals, χωρίς να έχει δηλωθεί, μέσω λογικού συμπεράσματος (reasoning) εφόσον οι δηλώσεις και οι κανόνες της Οντολογίας οδηγούν στο συμπέρασμα του ότι το Individual 1 είναι διαφορετικό από το Individual 2.

Τα παραπάνω συστατικά μιας οντολογίας δεν καλύπτονται από το RDF Schema μόνο, χρειάζεται μια **Γλώσσα Οντολογίας** με περισσότερες δυνατότητες έκφρασης, απαρίθμησης και διευκόλυνσης συμπερασμού.

2.5.2 Web Ontology Language

Όπως αναφέρθηκε στην παράγραφο 2.3.3 το RDFS είναι ουσιαστικά μια γλώσσα διατύπωσης οντολογιών με κάποιους περιορισμούς στην έκφραση των **Συστατικών μιας Οντολογίας**, όπως αυτά αναφέρονται στην παράγραφο 2.5.1.

Ζητώντας μεγαλύτερη εκφραστικότητα στις γλώσσες Οντολογίας διάφοροι οργανισμοί πήραν πρωτοβουλίες. Το Defense Advanced Research Projects Agency (DARPA) των ΗΠΑ παρουσίασε την γλώσσα **DAML[30]** (DARPA Agent Markup Language), ενώ στην άλλη πλευρά του Ατλαντικού παρουσιάστηκε η γλώσσα OIL (Ontology Inference Layer) από τους Dieter Fensel, Frank van Harmelen και Ian Horrocks. Και οι δύο γλώσσες βασίζονται στο RDF και στο RDFS, με την συνεργασία της DARPA και την χρηματοδότηση από το Information Society Technologies (IST) της ΕΕ προέκυψε η **DAML+OIL[31]**.

Η εξέλιξη της DAML+OIL είναι Web Ontology Language **OWL[32]** και σήμερα ο οργανισμός W3C την ορίζει ως τρείς υπογλώσσες που η κάθε μία καλύπτει μέρος των αναγκών του **ΣΙ**.

2.5.2.1 OWL Lite

Ο αρχικός σκοπός της ήταν να παρέχει μια απλή ιεραρχία με απλούς περιορισμούς, με την ελπίδα ότι θα αποτελέσει, λόγω της απλότητας της, επιλογή για migration από άλλα συστήματα. Πρακτικά όμως αποδείχθηκε ότι είναι εξίσου δύσκολη η ανάπτυξη εργαλείων όσο και με τις συντακτικά πιο εκφραστικές εκδόσεις OWL.

2.5.2.2 OWL DL

Σχεδιάστηκε να παρέχει εκφραστικότητα σε μέγιστο βαθμό, με υπολογιστική πληρότητα η οποία εγγυάται αποδεκτά αποτελέσματα και αποφυγή αοριστίας, μέσω αλγορίθμων, σε υπολογισμούς. Το Description Logic είναι μέρος του ονόματος που ανταποκρίνεται σε ένα θεμελιώδες χαρακτηριστικό της δομής της OWL, έτσι περιλαμβάνει όλα τα συστατικά που χρειάζεται, μια γλώσσα OWL, με κάποιους περιορισμούς.

2.5.2.3 OWL Full

Σχεδιάστηκε υπό διαφορετική οντολογική προσέγγιση από τις DL και Lite, με χαλαρότερη αλλά υπαρκτή συμβατότητα προς το DRF Schema, δεν παρέχει υπολογιστική πληρότητα, και επιτρέπει αοριστία στα αποτελέσματα, πράγμα που δίνει στο λογισμικό συλλογισμού (Reasoning Software) την δυνατότητα πλήρους ανάλυσης. Εκφράζει όλα τα συστατικά που χρειάζεται μια OWL γλώσσα,

επιτρέπει τον εμπλουτισμό του δεδομένου RDF λεξιλογίου και επιτρέπει τον ορισμό ιδιοτήτων σε συστατικά που η LIte και DL δεν επιτρέπουν.

2.5.3 Δομή της οντολογίας

Η δομή μιας Οντολογίας αποτελείται από:

Κεφαλίδα – Header

Είναι η ρίζα της οντολογίας και ορίζεται από ένα στοιχείο **rfd:DRF**, όπως περιγράφεται στην **2.3.2** παράγραφο, εκεί δηλώνονται τα tag αλλά και ο χώρος ονομάτων σαν prefix, ουσιαστικός παράγοντας η επιλογή tags και ονομάτων με μια λογική Human Readable για την ευκολία ανάπτυξης και συντήρησης.

Δηλώσεις Κλάσεων

Οι κλάσεις δηλώνονται από ένα στοιχείο **owl:Class** και όπως περιγράφεται στην παράγραφο **2.5.1**, υπάρχουν στοιχεία που δηλώνουν υποκλάσεις, ξένες ή ισοδύναμες κλάσεις.

Δηλώσεις ιδιοτήτων Αντικειμένων και Τύπων Δεδομένων

Οι ιδιότητες δηλώνονται με στοιχεία που περιγράφονται στις παραγράφους **2.3.3** και **2.5.1** με το **RDF Schema** και **OWL** σύνταξη αντίστοιχα. Για τα αντικείμενα ορίζονται οι ιεραρχικές σχέσεις μεταξύ τους. Συγκριτικές σχέσεις (ισότητα, ανισότητα, αντιστροφή), δηλώσεις πεδίων ορισμού και εφαρμογής (Range & Domain) αλλά και περιορισμοί στην πληθικότητα των υλοποιήσεων τους ορίζονται και για τα Δεδομένα αλλά και για τα Αντικείμενα.

Υλοποίηση Individual

Είναι το σημείο που **εισάγονται οντότητες**, οι οντότητες αυτές έχουν ιδιότητες που δηλώθηκαν και περιγράφονται στις προηγούμενες δηλώσεις (δηλώσεις κλάσεων, δηλώσεις Ιδιοτήτων) έτσι ορίζεται το είδος τους και ανάλογα με το είδος τους ακολουθούν οι περιορισμοί και οι σχέσεις τους. Το όνομα της οντότητας δεν την καθορίζει στο να είναι μοναδική ή διαφορετική ή ίδια με μια άλλη.

Βάσει πειραματισμού που έγινε σε αυτή την έρευνα διαπιστώθηκε η **επαγωγική λογική** του " *If it looks like a duck, swims like a duck, and quacks like a duck, then it probably is a duck.*". Αποφυγή του παραπάνω λογισμού γίνεται με ιδιότητες, της παραγράφου **2.5.1**, όπως DiferentIndividuals ή NegativeObjectPropertyAssertion για αποφυγή επαγωγικού συμπεράσματος σχέσης μεταξύ δυο Individual.

2.6 Γλώσσα SPARQL

Σε αυτό το σημείο συναντώνται στοιχεία προηγουμένων κεφαλαίων, αναλύονται όροι και έννοιες, που έμειναν εκτός μέχρι αυτό το σημείο για να αναλυθούν τώρα, λόγω της κοινής τους εφαρμογής σε όλη την προηγούμενη ανάλυση.

2.6.1 Γενικά

Η **SPARQL** είναι μια γλώσσα ερωτημάτων σε δεδομένα που είναι αποθηκευμένα σε μορφή **RDF**, η ονομασία της ακολουθεί την, ανέλπιστα συχνή, πρακτική στον κόσμο των Η/Υ του "αναδρομικού ακρωνυμίου" της φράσης που την περιγράφει: " **SPARQL Protocol And Rdf Query Language**".

Τα δεδομένα στα οποία γίνεται η αναζήτηση μπορούν να είναι και δεδομένα που δεν είναι αποθηκευμένα ως RDF αλλά μπορούν να εκφραστούν, μέσω ενδιάμεσων προγραμμάτων, σαν δεδομένα μορφής RDF. Έχει ορισθεί ως Πρότυπο από το RDF Data Access Working Group (**DAWG**) του World Wide Web Consortium (**W3C**), αναγνωρίστηκε σαν μια τεχνολογία "κλειδί" για τον Σημασιολογικό Ιστό και η έκδοση 1.1 από τον Μάρτιο του 2013**[33]** προτείνεται η χρήση της από το W3C. Αποτελεί εξέλιξη, προηγούμενων γλωσσών ερωτημάτων σε δεδομένα RDF, των **RDQL** και **SeRQL** και ανήκει και αυτή στο Query Language Paradigm. Η ομοιότητα της με την **SQL** είναι εμφανής και επιτηδευμένη, επενδύοντας στα δημοφιλή και ευρέως αποδεκτά χαρακτηριστικά της SQL, μιας γλώσσας που ορίζει το πλαίσιο, όσο καμία άλλη, των Query γλωσσών.

2.6.2 Σύνταξη Turtle

Η **Terse RDF Triple Language** (Turtle) εκτός από τρόπος σύνταξης για έκφραση δεδομένων **RDF[34]** είναι και πρότυπο αποθήκευσης αρχείου δεδομένων. Επιλέχθηκε μετά από πειραματισμό για την υλοποίηση των γράφων της ΔΕ, λόγω της απλότητας της σε σχέση με την χρήση N-Triplets, JSON-LD και RDF/XML αλλά βασικότερα λόγω της ομοιότητας με την SPARQL.

Η κάθε τριάδα πληροφορίας όπως το παράδειγμα της παραγράφου **2.3.2**, έχει :

Υποκείμενο Κατηγόρημα Αντικείμενο

Με την σύνταξη Turtle, εστιάζοντας μόνο στα **κοινά στοιχεία με την SPARL**, μια πλήρη τριάδα ορίζεται σαν:

Subject	**Predicate**	**Object**
my:Elephant	*base:titleEn*	*"Elephant".*

Και τελειώνει με τελεία (.)

Ενώ αν υπάρχει **επανάληψη του Υποκειμένου** η πρώτη τριάδα είναι πλήρης και τελειώνει με «;», και οι επόμενες τριάδες είναι δυάδες αφού νοείται επανάληψη του Υποκειμένου, η τελευταία δυάδα τελειώνει με τελεία:

Subject	**Predicate**	**Object**
my:Elephant	*base:titleEn*	*"Elephant";*
	base:titleEl	*"Ελέφαντας".*

Και αν υπάρχει **επανάληψη του Υποκειμένου και του Κατηγορήματος** η πρώτη τριάδα είναι πλήρης και τελειώνει με comma και οι επόμενες τριάδες είναι ουσιαστικά μια λίστα μονάδων αφού υπάρχει επανάληψη του Υποκειμένου και του Κατηγορήματος, η τελευταία μονάδα τελειώνει με τελεία:

Subject	**Predicate**	**Object**
my:Elephant	base:desc	"Elephant",
		"Elephantidae",
		"Mastodont".

2.6.3 Δομή SPARQL

Υπάρχουν σταθερά δομικά στοιχεία που είναι ξεκάθαρα ορισμένα και, όπως και σε άλλες γλώσσες, ορίζουν σταθερές, μεταβλητές, σχόλια και υπάρχει εκφραστική σύνταξη.

Τα βασικά δομικά στοιχεία είναι :

Comments # Τα σχόλια ξεκινάν με '#' μέχρι το τέλος της γραμμής και δεν επηρεάζουν σε καμία των περιπτώσεων την εκτέλεση των εντολών.

URI

Uniform Resource Identifier, είναι μια σειρά χαρακτήρων που ορίζει μοναδικά ένα πόρο, που χρησιμοποιείται από τεχνολογίες του web.

Πλήρες URI < http://www.allios.co.nf/Person#Yiannis> Είναι ένα πόρος

Με πρόθεμα :

PREFIX my: < http://www.allios.co.nf/Person#>

Δηλαδή my:Yiannis

Αντί για : < http://www.allios.co.nf/Person#Yiannis>

Literals

Πρόκειται για αντικείμενα, είναι υλοποίηση κάποιου τύπου δεδομένων βάσει του XML/RDF και OWL λεξιλογίου.

"*Plain literal*"

"*Δεν κάνει κρύο στην Ελλάδα*"@el Είναι literal με tag γλώσσας.

"*2021*"^^xsd:integer Είναι literal τύπου integer.

"true"^^xsd:Boolean Είναι literal λογικού τύπο.

"*4.2*"^^xsd:decimal Είναι literal τύπου decimal.

Variables

Πρόκειται για μεταβλητές, με όλα όσα συνεπάγονται από την έννοια της μεταβλητής στον κόσμο της πληροφορικής, με την μόνη διαφορά ότι το περιεχόμενο είναι ένα συστατικό μιας οντολογικής τριάδας όπως παρουσιάστηκε στο 2.6.2.

?variable, ?μεταβλητή, ?καιΑλληΜια

Triple Patterns

Πρόκειται για μοτίβο τριάδας, που ταιριάζει σε όλα ή δύο ή ένα στοιχείο μιας τριάδας.

my:Elephant base:titleEn "Elephant" μία και μοναδική τριάδα χωρίς μεταβλητή.

?μεταβλητή base:desc " Mastodont " Το περιεχόμενο της μεταβλητής **?μεταβλητή** θα είναι **Υποκείμενα** που έχουν Κατηγόρημα "base:desc" και Αντικείμενο "Mastodont".

my:Elephant ?katigorima ?timi Οι μεταβλητές θα πάρουν αντίστοιχα τα περιεχόμενα των τριάδων που έχουν Υποκείμενο το " ***my:Elephant",*** όλες οι τιμές της "?katigorima" θα είναι

Κατηγορήματα… όχι λόγω ονόματος αλλά λόγω θέσης στην τριάδα, αντίστοιχα τα περιεχόμενα της "?timi" θα είναι όλα Αντικείμενα.

Τέσσερα είδη SPARQL Queries

- SELECT
  - Για την εμφάνιση συγκεκριμένων μεταβλητών και εκφράσεων.
- ASK
  - Πρόκειται για ερώτηση αν υπάρχουν αποτελέσματα σε ερώτημα με αποτέλεσμα Boolean.
- DESCRIBE
  - Παρουσίαση σε RDF τριάδες όπου εμπλέκεται η μεταβλητή του ερωτήματος.
- CONSTRUCT
  - Για κατασκευή RDF τριάδων ή γράφων βάσει ερωτήματος και μεταβλητών.

2.6.4 Ερώτημα SPARQL

Με αφαιρετικό τρόπο περιγράφεται η δομή ενός απλού ερωτήματος :

PERFIX ....

SELECT …

FROM …

WHERE {…}

GROUP BY …

Η γενική περιγραφή κάθε μέρους:

Prefix declarations, δηλώσεις προθέματος όπως περιγράφονται στο 2.6.3

Με την δήλωση *@prefix foo:<http://www.megalo_onoma.gr/mantsou/>*

αντί να δηλώνεται στο Query το URI ως ***<http://www.megalo_onoma.gr/mantsou/#bar>***

το δηλώνεται απλά σαν ***foo:bar***.

Result Clause, ποιες πληροφορίες θα δώσει το ερώτημα.

Δηλαδή : *SELECT ?name* όπου το "**?name**" είναι μια μεταβλητή.

Dataset definition, Η δήλωση ποιών RDF γράφων χρειάζονται για το ερώτημα

FROM foo:graph1

Query Pattern, ορίζεται το μοτίβο που αναζητείται στα δεδομένα

WHERE { ?name foo:aSonOf foo:mantsu }

Query Modifiers, δήλωση για ταξινόμηση, περικοπή και τακτοποίηση του αποτελέσματος

ORDER BY ?name

Φυσικά λόγω της δομής των δεδομένων (τριάδες) πάντα γίνεται αναφορά σε σχέση με ισχύουσα τριάδα ή ισχύουσες τριάδες και τα δεδομένα προς εμφάνιση είναι μέρος μιας τριάδας ή σύνθεσης τριάδων. Ακολουθώντας τα δυνατά στοιχεία της SQL απεικονίζονται **ενώσεις, τομές, διαφορές** και **περιορισμοί:**

Τομή:

SELECT δεδομένα προς εμφάνιση

WHERE

{

μια ισχύουσα τριάδα .

άλλη ισχύουσα τριάδα

}

Το αποτέλεσμα αποτελεί την **Τομή** των δεδομένων που επαληθεύουν ως αληθείς και τις δύο τριάδες.

Τομή OPTIONAL:

SELECT δεδομένα προς εμφάνιση

WHERE

{

μια ισχύουσα τριάδα .

άλλη μια ισχύουσα τριάδα

OPTIONAL{Επιπλέον ισχύουσα τριάδα}

}

Το αποτέλεσμα αποτελεί την **Τομή** των δεδομένων που επαληθεύουν ως αληθείς και τις τρεις τριάδες αλλά εφόσον, και αν, **υπάρχει αληθής επιπλέον τριάδα**, η μη εύρεση αληθούς τρίτης τριάδας (στο OPTIONAL μέρος) δεν φέρει καμία διαφορά στην εκτέλεση του ερωτήματος.

Ένωση :

SELECT δεδομένα προς εμφάνιση

WHERE

{

{ μια ισχύουσα τριάδα.}

UNION

{ άλλη μια ισχύουσα τριάδα.}

}

Το αποτέλεσμα αποτελεί την **Ένωση** των δεδομένων που επαληθεύουν ως αληθείς τις δύο τριάδες.

Διαφορά :

SELECT δεδομένα προς εμφάνιση

WHERE

{

{ μια ισχύουσα τριάδα.}

MINUS

{ άλλη μια ισχύουσα τριάδα.}

}

Το αποτέλεσμα είναι η **Διαφορά** του συνόλου των δεδομένων που επαληθεύουν την μία τριάδα και του συνόλου των δεδομένων που επαληθεύουν την άλλη τριάδα, εφόσον και αν υπάρχουν αποτελέσματα της άλλης τριάδας και αυτά είναι υποσύνολο των αποτελεσμάτων της πρώτης.

Περιορισμοί :

SELECT δεδομένα προς εμφάνιση

WHERE

{

ισχύουσα τριάδα.

*FILTER (μια αληθής **Συνθήκη** σε σχέση με τα δεδομένα προς εμφάνιση)*

}

Το αποτέλεσμα είναι το σύνολο των δεδομένων που επαληθεύουν την τριάδα αλλά χωρίς το υποσύνολο των δεδομένων που δεν ικανοποιούν την συνθήκη που τέθηκε.

Αξίζει να τονισθεί ότι στην **Συνθήκη** εκτός απο τους γνωστούς τελεστές (>=, <=, ==, !=, &&, ||) διατίθενται και ειδικοί λογικοί τελεστές όπως "**isURI**", "**isLiteral**" και "**isBlank**" που μπορούν να χρησιμοποιηθούν για το "φιλτράρισμα" των δεδομένων, πέρα από τους τελεστές δίδεται και η χρήση Regular Expressions μορφής ***REGEX(?μεταβλητη, "έκφραση").***

Ομαδοποίηση - Ταξινόμηση :

SELECT ?μεταβλητή1 (COUNT(?μεταβλητή2) AS ?εμφανίσεις)

WHERE { ?μεταβλητη2 οτιδήποτε ?μεταβλητη1.}

GROUP BY (?μεταβλητή1)

ORDER BY DESC (?εμφανίσεις)

Το αποτέλεσμα εμφανίζει ομαδοποιημένα την ***?μεταβλητή1*** και την μεταβλητή ?εμφανίσεις που μετρά τις επαναλήψεις της ***?μεταβλητή2*** ανα ***?μεταβλητή1*** κατά φθίνουσα ταξινόμηση.

2.6.5 SPARQL Update, Delete, Insert

Όπως αναφέρθηκε αναλυτικά στα ερωτήματα SPARQL ο όρος "WHERE" καθορίζει τις συνθήκες που θα εφαρμοστεί το "SELECT", με τον ίδιο τρόπο καθορίζεται η συνθήκη για την εισαγωγή ή την διαγραφή, όπου στην θέση του "SELECT" καλείται ο όρος "INSERT" 'η "DELETE" αντίστοιχα.

Δομή ενός απλού Insert :

PERFIX ....

INSERT { τριάδα ή τριάδες }

WHERE { ισχύουσα συνθήκη}

Δομή ενός απλού Delete :

PERFIX ....

DELETE { τριάδα ή τριάδες }

WHERE { ισχύουσα συνθήκη}

Η δομή ενός Update είναι ένα Delete με ένα Insert όπου ισχύει το κοινό Where:

PERFIX ....

DELETE { τριάδα ή τριάδες}

INSERT { τριάδα ή τριάδες }

WHERE { συνθήκη με συσχετισμό μέσω μεταβλητής με τις προηγούμενες δηλώσεις}

2.7 Βιβλιοθήκη Apache Jena

Το Apache Jena είναι ένα προγραμματιστικό πλαίσιο (Framework) Σημασιολογικού Ιστού, φτιαγμένο για την java που παρέχει διεπαφή προγραμματισμού εφαρμογών (API) για την λήψη και αποστολή δεδομένων σε RDF γράφο (graph). Παρόμοιο μεν με το OpenRDF Sesame **rdf4j [35]**, αλλά επιπλέον προσφέρει υποστήριξη για OWL (Web Ontology Language). Παρέχει την δυνατότητα υλοποίησης Reasoner και ένας από αυτούς είναι ο **Pellet[36]**, μία Java OWL-DL μηχανή εξαγωγής συμπερασμάτων, ανοικτού κώδικα. Το πακέτο της Jena περιέχει interfaces για την αναπαράσταση μοντέλων, πόρων, ιδιοτήτων, literals και δηλώσεων όλα σύμφωνα με την λογική του RDF γράφου. Επί της ουσίας κάνοντας εισαγωγή την βιβλιοθήκη jena σε μια εφαρμογή Java αποκτά ο προγραμματιστής της εφαρμογής πρόσβαση μέσω διαδικασιών, τύπων και εργαλείων, τα λεγόμενα Factories, σε Σημασιολογικές Έννοιες, τα βασικά πακέτα είναι**[37]**:

Πίνακας 2.2: Βασικά πακέτα Jena

Package	Description	More information
oaj.jena.rdf.model	The Jena core. Creating and manipulating RDF graphs.	
oaj.riot	Reading and Writing RDF.	
oaj.jena.datatypes	Provides the core interfaces through which datatypes are described to Jena.	Typed literals
oaj.jena.ontology	Abstractions and convenience classes for accessing and manipulating ontologies represented in RDF.	Ontology API
oaj.jena.rdf.listeners	Listening for changes to the statements in a model	
oaj.jena.reasoner	The reasoner subsystem is supports a range of inference engines which derive additional information from an RDF model	Reasoner how-to
oaj.jena.shared	Common utility classes	
oaj.jena.vocabulary	A package containing constant classes with predefined constant objects for classes and properties defined in well known vocabularies.	

oaj.jena.xmloutput | Writing RDF/XML. | I/O index

2.7.1 RDF API

Σε αυτή την διεπαφή χρήσης, όλες οι πληροφορίες που μπορεί να δώσει μια συλλογή τριάδων τύπου RDF περιέχονται σε μία δομή της Jena που λέγεται Model, ουσιαστικά πρόκειται για ένα Graph (Γράφο) που έχει κόμβους RDF μαζί με τις σχέσεις που τους διέπουν, όπως αναφέρθηκαν στο **2.3.2** και **2.3.3**.

Μιλώντας «Java» λέμε ότι δημιουργώντας ένα αντικείμενο τύπου (Κλάσης) Model έχουμε δημιουργήσει μία δομή η οποία περιέχει πληροφορίες ενός RDF γράφου. Η κλάση Model έχει πλούσιο API με σκοπό να κάνει εύκολη την ανάπτυξη εφαρμογών που βασίζονται σε RDF δομές, και RDFS γλώσσα. Παρέχει την αναγκαία γενικότητα πάνω από διαφορετικού είδους (εξειδικευμένους) τρόπους αποθήκευσης της RDF πληροφορίας, παρέχει προσωρινές δομές (in-memory) αλλά και δομές με αποθήκευση (persistent) και διάφορους τρόπους επικοινωνίας με αυτές.

2.7.2 Ontology API

Μέσω της διεπαφής χρήσης της OWL καλύπτονται όλα όσα μπορούν να γίνουν με RDFS αλλά και περισσότερα, όπως αναφέρθηκε στο **2.5.3**, η OWL καλύπτει περισσότερες οντολογικές έννοιες. Το Ontology API της Jena έχει κρατήσει αποστάσεις από το συντακτικό της εκάστοτε γλώσσας, δηλαδή η Java κλάση OntClass μπορεί να αντιπροσωπεύει τόσο μια OWL κλάση όσο και μια RDFS και για τις ιδιαιτερότητες κάθε γλώσσας υπάρχει το αντίστοιχο προφίλ που ορίζει ανάλογα τα επιτρεπόμενα στοιχεία. Το προφίλ ουσιαστικά εξειδικεύει το Model, χωρίς εξειδίκευση έχουμε το πρόσβαση σε μια «απλή» συλλογή RDF και με το προφίλ αποκτάμε την επιπλέον υποστήριξη για Κλάσεις Ιδιότητες και Individual που περιμένουμε να έχουμε σε μια Οντολογία.

Προσοχή! Δεν αλλάζει μορφή η πληροφορία, όλα παραμένουν σε τριάδες RDF αποθηκευμένα με το μοντέλο RDF, αυτό που γίνεται είναι η παροχή, στον προγραμματιστή της Java εφαρμογής, μεθόδων και λειτουργιών που κάνουν πιο εύκολη την διαχείριση των δεδομένων με έναν προγραμματιστικό τρόπο.

2.7.3 Μηχανισμός ARQ

Είναι η μηχανή ερωτημάτων (query) της Jena, υποστηρίζει την SPARQL που αναφέρεται στο κεφάλαιο **2.6**.

Το βασικό πακέτο για την Java είναι : org.apache.jena.query και περιέχει**[38]** :

Query

Περιέχει ότι αφορά το query σε ένα πρόγραμμα. Συνήθως καλείται μια από τις μεθόδους του QueryFactory για να δημιουργηθεί ένα τέτοιο αντικείμενο.

QueryExecution

Είναι το αντικείμενο που αντιπροσωπεύει την εκτέλεση του ερωτήματος.

QueryExecutionFactory

Είναι μέθοδος μέσω της οποίας μπορεί να υλοποιηθεί ένα αντικείμενο τύπου QueryExecution.

DatasetFactory

Μέθοδοι δημιουργίας συνόλων δεδομένων.

SELECT queries:

QuerySolution

Ένα αντικείμενο που παρέχει κάποιες πληροφορίες για τα αποτελέσματα, όπως ονόματα μεταβλητών κ.α. αλλά δίνει και πρόσβαση σε σε πόρους της δεδομένης απάντησης (λύσης).

ResultSet

Ένα σύνολο από απαντήσεις, που μπορούν να προσπελαστούν σαν πίνακας, με μεθόδους που δίνουν πληροφορίες για το μοντέλο, τις μεταβλητές, αν υπάρχει επομένη λύση κ.α.

ResultSetFormatter

Μετατρέπει τα αποτελέσματα σε διάφορες μορφές που επιθυμεί ο προγραμματιστής από απλή XML, text, json κα.

2.7.4 Fuseki server

Είναι ο SPARQL server της jena, μπορεί να σταθεί σε ένα απλό λειτουργικό σύστημα (όχι server) αλλά και σαν Java Web Application (WAR). Σε κάθε περίπτωση ακολουθείται η λογική της Apache και απαιτεί ακόμα και σε stand alone εγκαταστάσεις να τρέχουν οι απαραίτητες, σαν server, υπηρεσίες.

2.8 JavaFX

Η JavaFx είναι μια πλατφόρμα ανάπτυξης desktop και web εφαρμογών για Windows, Windows Mobile, Linux, OpenSolaris, macOS κτλ. Δεν αντικαθιστά το Swing και το AWT αλλά από την έκδοση 11 του JDK το 2018 είναι μέρος του Open Java Development Kit (OpenJDK). Παρέχει διεπαφή προγραμματισμού εφαρμογών (API) η οποία είναι προσαρμοσμένη για εφαρμογές desktop ή για mobile εφαρμογές. Απο την δεύτερη έκδοση της**[39]** δεν υποστηρίζει OpenSolaris και τις mobile εφαρμογές, αλλά, σημαντικότερα, **δεν μας υποχρεώνει να χρησιμοποιούμε JavaFX Script** κάνοντας τον κώδικα JavaFx να είναι δυσδιάκριτος απο τον κώδικα της Java, επίσης γίνεται χρήση νέας "Δηλωτικής" γλώσσας XML που καλείται FXML. Με την έκδοση JavaFX2.1 παρουσιάστηκε το εργαλείο JavaFX Scsene Builder για την Οπτική Σχεδίαση των διεπαφών.

Αυτά τα χαρακτηριστικά έχουν θεωρηθεί σαν πλεονέκτημα για τις ανάγκες αυτής της έρευνας και οδήγησαν στην επιλογή της για την σχεδίαση και υλοποίηση της διεπαφής χρήστη.

2.8.1 FXML

Δίνει την υποδομή να δηλώνονται σε μορφή τύπου XML, όπως αναφέρθηκε στο 2.3.1, τα γραφικά στοιχεία μιας εφαρμογής, χωρίζοντας το προγραμματιστικό μέρος, από το μέρος της Παρουσίασης. Τα στοιχεία (αντικείμενα) που δηλώθηκαν και υλοποιήθηκαν από την FXML είναι πλήρως προσβάσιμα, επεξεργάσιμα και διαχειρίσιμα από την εφαρμογή και η συμπεριφορά τους είναι σαν να έχουν δημιουργηθεί από κώδικα java εξ αρχής.

Θεωρήθηκε ως πλεονέκτημα κατά την παρούσα έρευνα η δημιουργία της διεπαφής σε μορφή τύπου XML μια και είναι ευκολότερη η κατανόηση, η αλλαγή της οπτικής δομής, αλλά και η τροποποίηση

της διεπαφής ενώ είναι σε εξέλιξη ή ακόμα και ολοκληρωμένο το αντίστοιχο προγραμματιστικό μέρος της εφαρμογής.

Η FXML δεν "*γίνεται compile*" μαζί με τον κώδικα, δηλαδή αν γίνουν αλλαγές στο fxml αρχείο δεν χρειάζεται να επαναλαμβάνεται το compile όλης της εφαρμογής.

Παράδειγμα δήλωσης FXML

```
<BorderPane>
        <top>   <Label text="Page Title"/>      </top>
        <center>        <Label text="Some data here"/>          </center>
</BorderPane>
```

Με τα παραπάνω αναγνωρίζει και δημιουργεί η JavaFX ενα αντικείμενο BorderPane που περιέχει δύο αντικείμενα τύπου Label και ορίζει την θέση τους και τα περιεχόμενα τους. Η υλοποίηση με την λογική "Γονέας – Παιδί" των στοιχείων Διεπαφής Χρήστη εκτός από την καλή οργάνωση προσφέρει και ευελιξία στην λήψη πληροφορίας από τον προγραμματιστή καθώς δεν χρειάζεται να θυμάται τα ID του γονέα ενός κόμβου ούτε τα ID των "Αδελφών" του. Μπορεί να περιηγηθεί στο "XML-Δένδρο" και να πάρει ή να δώσει στοιχεία στην Διεπαφή χάρη στα εργαλεία του API Της JavaFX.

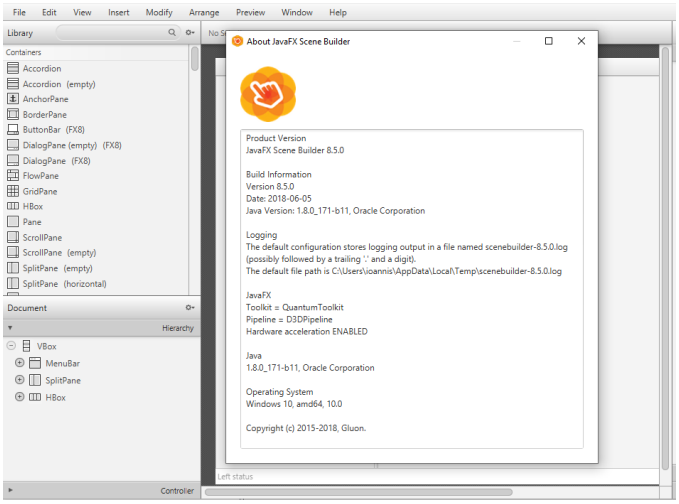
2.8.2 Scene Builder

Είναι μια εφαρμογή συμβατή με το "οικοσύστημα" της JavaFX. Στην *Εικόνα 2.4* παρουσιάζεται η έκδοση 8.5.0 που χρησιμοποιήθηκε σε αυτή την ΔΕ.

Εργαλείο οπτικής σχεδίασης που επιτρέπει στους προγραμματιστές να σχεδιάζουν γρήγορα, και χωρίς κόπο, διεπαφές χρήστη για το API της JavaFX.

Δεν απαιτείται να γράψουν κώδικα και ορίζονται όλα τα χαρακτηριστικά που χρειάζεται ο προγραμματιστής για να επέμβει με κώδικα, αν το θεωρεί αναγκαίο.

Το αποτέλεσμα είναι ένα αρχείο FXML αλλά και ο αντίστοιχος κώδικας Java (javaFX), μπορεί στη συνέχεια να γίνει μέρος μιας ολοκληρωμένης εφαρμογής Java υλοποιώντας το περιβάλλον εργασίας του χρήστη.



Εικόνα 2.4: JavaFX Scene Builder

Κεφάλαιο 3ο: Υλοποίηση συστήματος

Μέχρι στιγμής ερμηνεύτηκε το τι είναι μια εκδήλωση και επιλέχθηκαν τα πληροφοριακά σημαντικά στοιχεία της, σχεδιάστηκε ένα μοντέλο υλοποίησης, ερευνήθηκαν οι διαθέσιμες δομές δεδομένων και οι σημασιολογικά υλοποιήσιμες τεχνολογίες.

Γίνεται παρακάτω η οντολογική σχεδίαση, η υλοποίηση σαν Οντολογία και ο έλεγχος της. Σχεδιάζεται το ΠΣ, αναλύονται κομβικά σημεία του, γίνεται εισαγωγή δοκιμαστικών δεδομένων και εξάγεται μια οντολογία σε αρχείο η οποία δοκιμάζεται στο πρόγραμμα Protégé και στο endpoint Virtuoso.

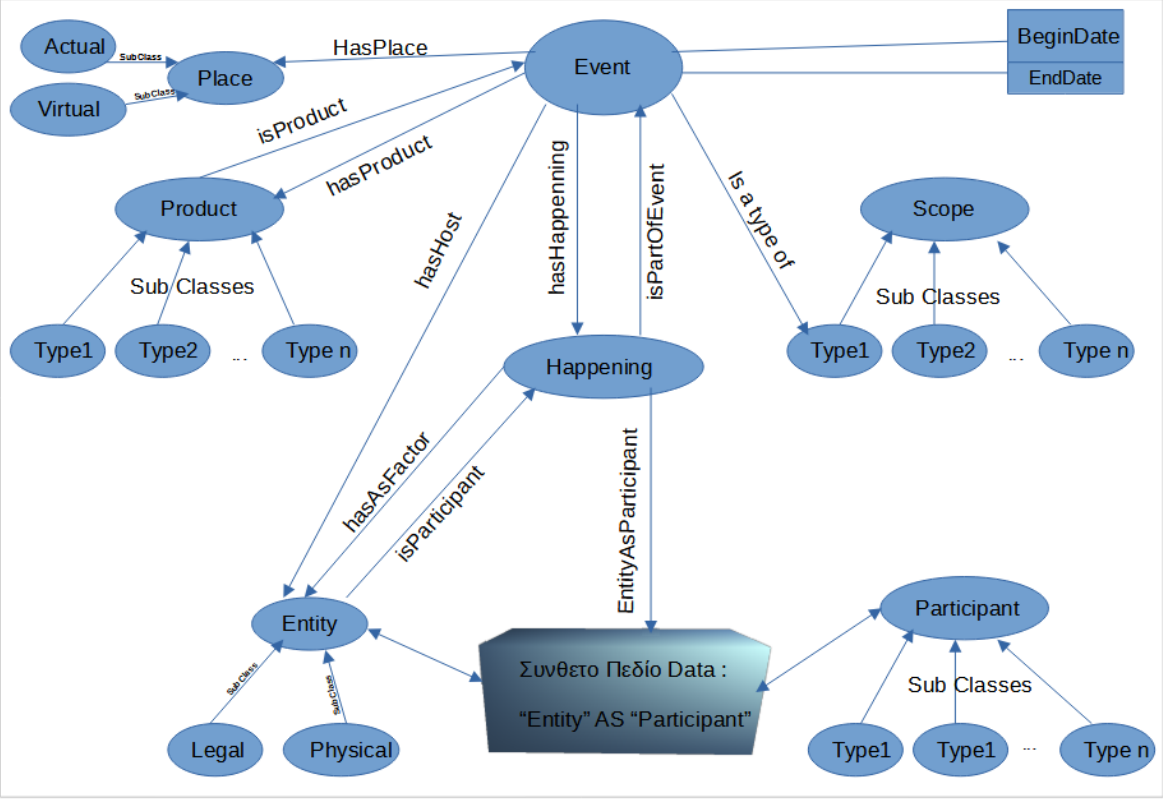
3.1 Δημιουργία οντολογίας

Επανασχεδιάζεται η δομή των κλάσεων του ΠΣ λαμβάνοντας υπ' όψη την οντολογική διάσταση, αποφασίζονται θέματα μοναδικότητας εγγραφών και ιεραρχίας.

Αποφασίστηκε να εκλείψουν τα πεδία «κλειδί» (ID) από την σχεδίαση της οντολογίας και να γίνει χρήση των **URI** μέσα στο ΠΣ αφού (θα πρέπει να) πληρούν τις ιδιότητες ενός αναγνωριστικού στοιχείου.

3.1.1 Τροποποίηση κλάσεων και σχέσεων

Μελετώντας την δομή του διαγράμματος του *Σχήματος 1.1* υλοποιείται ένα οντολογικό διάγραμμα που αναπαριστά τις Κλάσεις, τις Σχέσεις και τα πεδία δεδομένων που συμμετέχουν ενεργά στην δομή της οντολογίας στο *Σχήμα 3.1*.



Σχήμα 3.1: Διάγραμμα Πληροφοριακού Συστήματος

Το «**Σύνθετο Πεδίο Data**» οντολογικά είναι απλά ένα πεδίο String αλλά προγραμματιστικά αποτελεί μια **σύνθεση πληροφορίας**, η πληροφορία αυτή είναι:

« *Συμμετέχει ο **Entity Individual** ως **Participant Individual*** »

Συγκεκριμένα **Entity** Individual είναι μια υλοποίηση των κλάσεων Legal ή Physical και περιγράφει ένα άτομο ή έναν οργανισμό π.χ. ***«Γιάννης Παπαδόπουλος».***

Participant Individual είναι μια υλοποίηση των **υποκλάσεων** της κλάσης Participant (που έχει ορίσει ο χρήστης) και περιγράφει ένα είδος συμμετοχής π.χ. η υποκλάση Staff (ορισμένη από την χρήστη) και υλοποίηση της Staff σε έναν Individual ***«Τεχνικός Ήχου Εικόνας»*** και η υποκλάση Honored (ορισμένη από την χρήστη) και υλοποίηση της Honored σε έναν Individual ***«Βραβευόμενος Εθελοντής».***

Με αυτόν τον τρόπο αποθηκεύεται το String «***Γιάννης Παπαδόπουλος ως Τεχνικός Ήχου Εικόνας»*** αλλά και το String «***Γιάννης Παπαδόπουλος ως Βραβευόμενος Εθελοντής»*** σαν πληροφορίες ενός συγκεκριμένου Happening Individual.

Αν και υπάρχει η σχέση “hasFactor” και “isParticipant” που συνδέει έναν **Entity Individual** με ένα **Happening Individual**, δεν περιγράφεται το είδος της σχέσης αλλά και αν περιγράφονταν (Οντολογικά) θα περιορίζονταν σε μία είδους σχέση.

Με χρήση κώδικα εξασφαλίζεται η λήψη από τα String τύπου “EntityAsParticipant” το μέρος **αριστερά** των χαρακτήρων «*ως*» και αντιμετωπίζεται σαν **Entity Individual** και το εκ των **δεξιά** μέρος των χαρακτήρων «*ως*» και αντιμετωπίζεται σαν **Participant Individual** με την Οντολογική τους σημασία.

Οι κλάσεις παραμένουν και είναι:

- Event
- Place
- Actual
- Virtual
- Product (abstract)
    - Υποκλάσεις ορισμένες από τον χρήστη
- Happening
- Scope (abstract)
    - Υποκλάσεις ορισμένες από τον χρήστη
- Entity
- Legal
- Physical
- Participant (abstract)
    - Υποκλάσεις ορισμένες από τον χρήστη

IsATypeOf: Ένα Individual τύπου **Event**, θα δηλώνεται **και σαν τύπου Scope** για ομαδοποίηση των εκδηλώσεων ανά σκοπό μέσω κλάσεων.

Οι σχέσεις Κλάσεων (Object Properties) είναι:

- isProduct (Είναι Product ενός Event)
- hasProduct (Το Event έχει κάποιο Product)
- hasHappening (Το Event έχει κάποιο Happening)

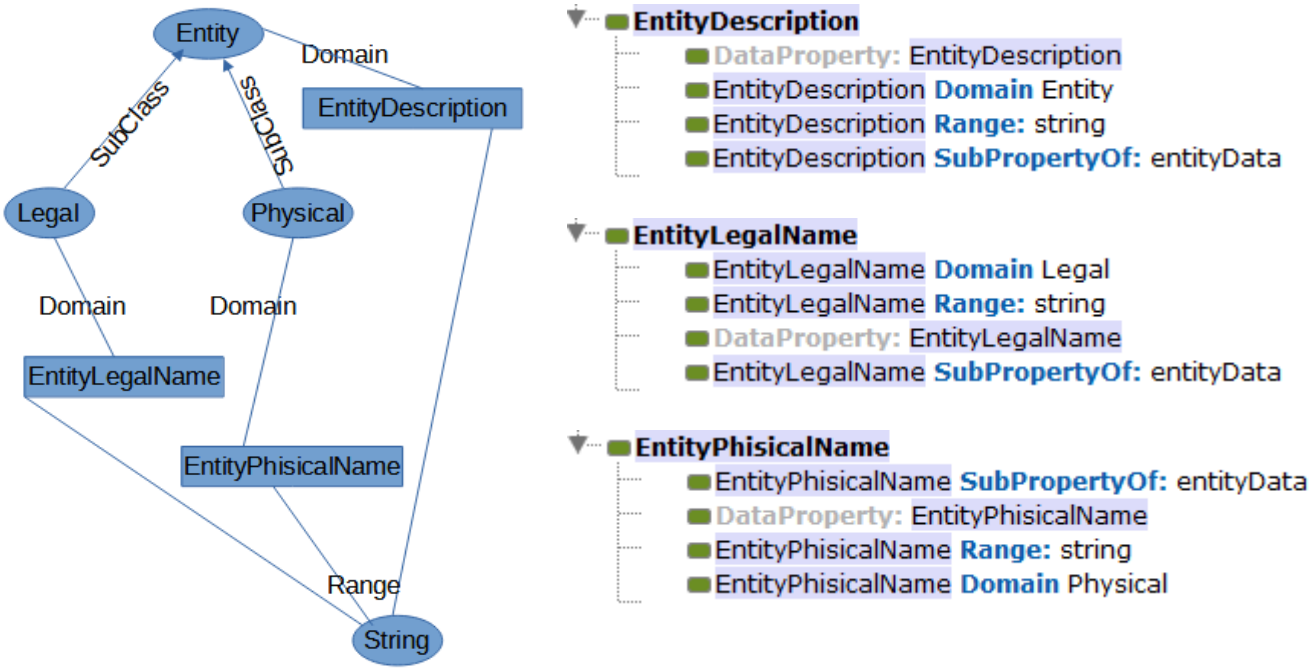
- isPartOfEvent (Το Happening ανήκει σε Event)
- isParticipant (Το Entity συμμετέχει σε ένα Happening)
- hasAsFactor (Το Happening έχει ένα Entity να συμμετέχει)
- hasHost (Το Event έχει ένα Entity διοργανωτή)
- hasPlace (Το Event έχει ένα Place που συμβαίνει)

Τα πεδία δεδομένων αλλά και οι πληθικές σχέσεις μεταξύ των συστατικών παρουσιάζονται στην επόμενη παράγραφο.

3.1.2 Υλοποίηση κλάσεων και Data properties στο Protégé

Καταχωρήθηκαν οι κλάσεις και τα Data Properties βάσει του σχεδιασμού του *σχήματος 3.1*.

Class Entity:

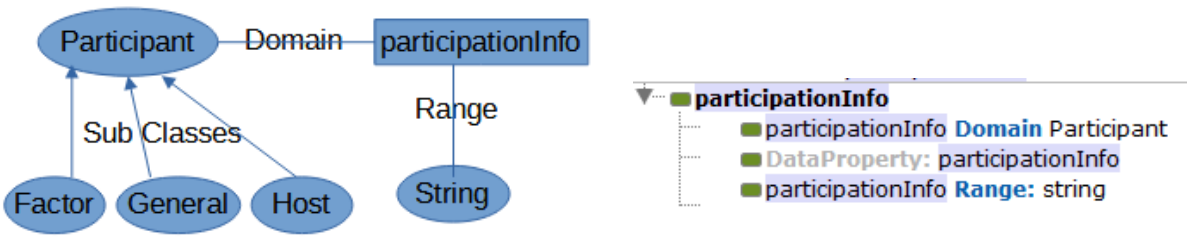


Σχήμα 3.2: Διάγραμμα κλάσης Entity

Αναπαριστώνται στο *Σχήμα 3.2* τα πεδία δεδομένων που θα χρησιμοποιηθούν και απεικονίζονται τα Range και Domain τους και δίδεται περιγραφή προερχόμενη από το Protégé.

Το πεδίο EntityDescription είναι κοινό και η κλάση Entity δεν θα είναι υλοποιήσιμη.

Class Participant:



Σχήμα 3.3: Διάγραμμα κλάσης Participant

Όπως φαίνεται στο *Σχήμα 3.3*, δημιουργήθηκαν τρείς κλάσεις, για δοκιμή και έλεγχο:

Factor: ως παράγοντας συμμετοχής.

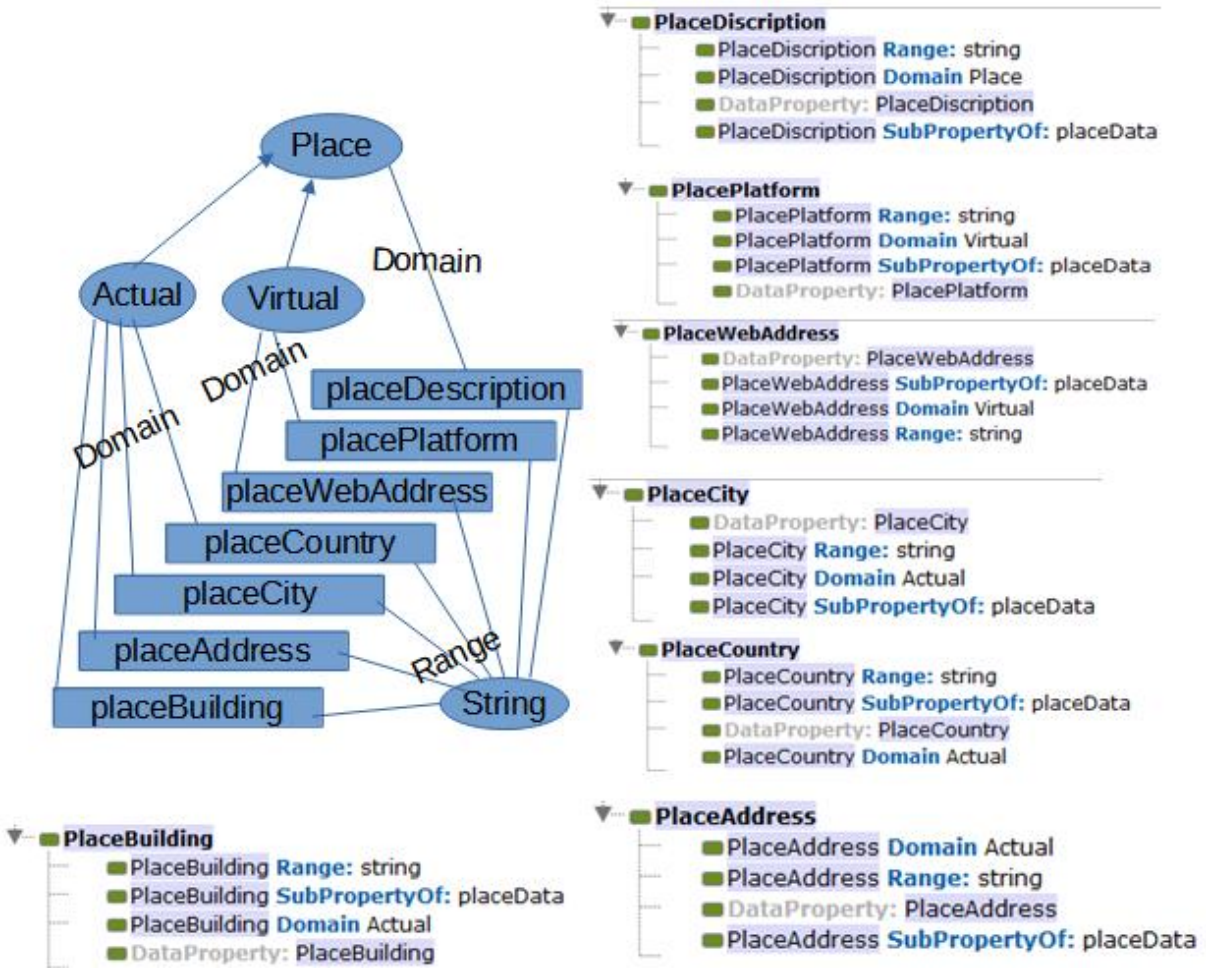
General: ως συμμετέχοντα γενικά.

Host: ως συμμετέχοντα εκ του διοργανωτή.

Η κλάση Participant θα παραμείνει μη υλοποιήσιμη στο ΠΣ και θα δίδεται η δυνατότητα καταχώρησης υποκλάσεων στον χρήστη για κατηγοριοποίηση του τρόπου συμμετοχής των Individual σε Happening.

Class Place:

Στο *Σχήμα 3.4* παρουσιάζεται η δομή της κλάσης Place με τα δεδομένα και τις σχέσεις της και δίδεται περιγραφή από το Protégé.

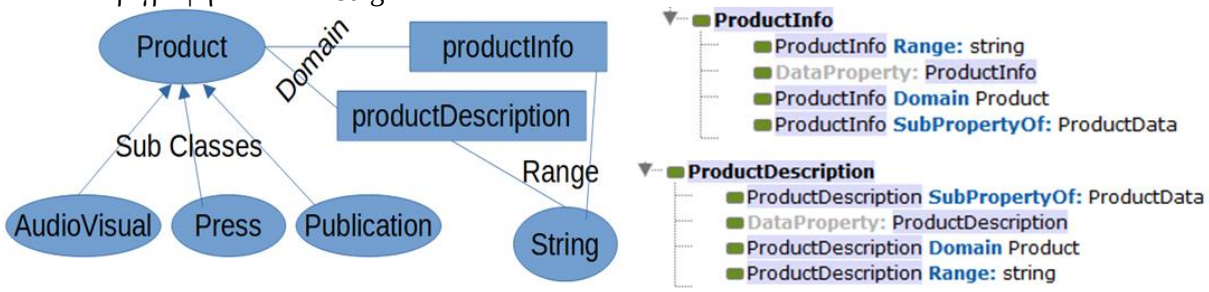


Σχήμα 3.4: Διάγραμμα κλάσης Place

Η κλάση Place δεν θα είναι υλοποιήσιμη κλάση αλλά θα χρησιμοποιηθεί από το ΠΣ για περιπτώσεις αναζήτησης Individual χωρίς να έχει σημασία αν είναι Actual ή Virtual με την χρήση της ιδιότητας **subClassOff**.

Class Product:

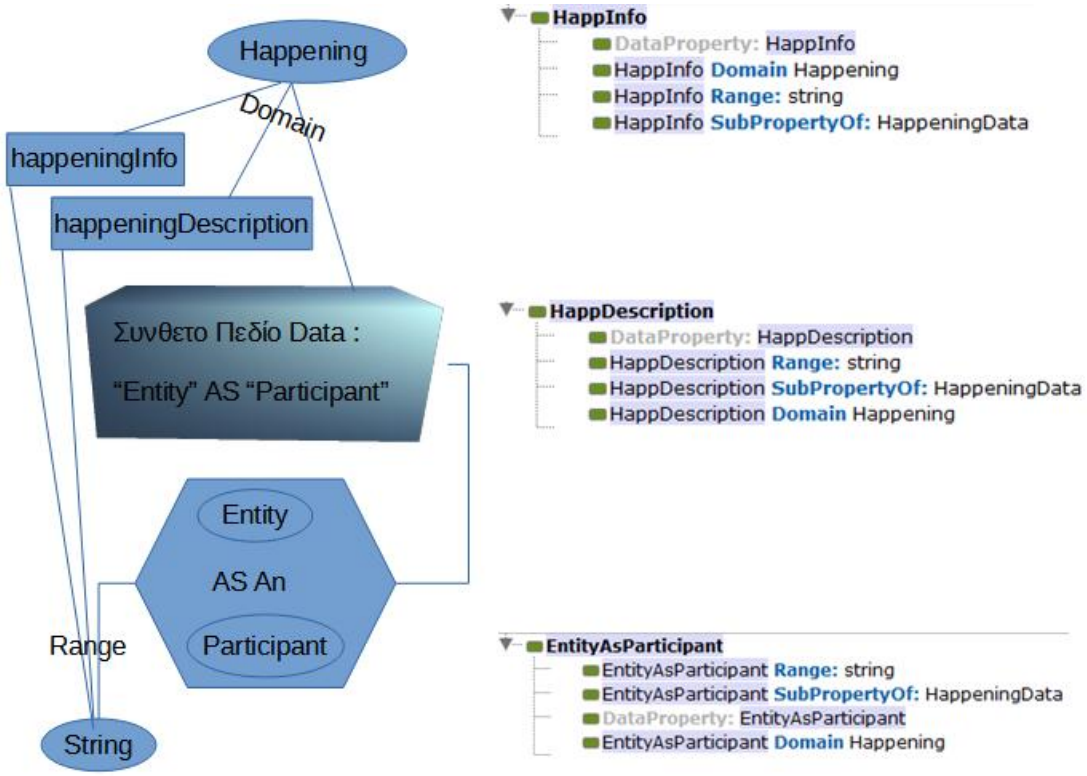
Στο *Σχήμα 3.5* παρουσιάζεται η δομή της κλάσης Product με τα δεδομένα και τις σχέσεις της και δίδεται περιγραφή από το Protégé.



Σχήμα 3.5: Διάγραμμα κλάσης Product

Υπάρχουν δύο πεδία δεδομένων στο επίπεδο της υπερκλάσης Product η οποία και αυτή δεν θα είναι υλοποιήσιμη και θα δίδεται η δυνατότητα στον χρήστη του ΠΣ να δημιουργεί υποκλάσεις της.

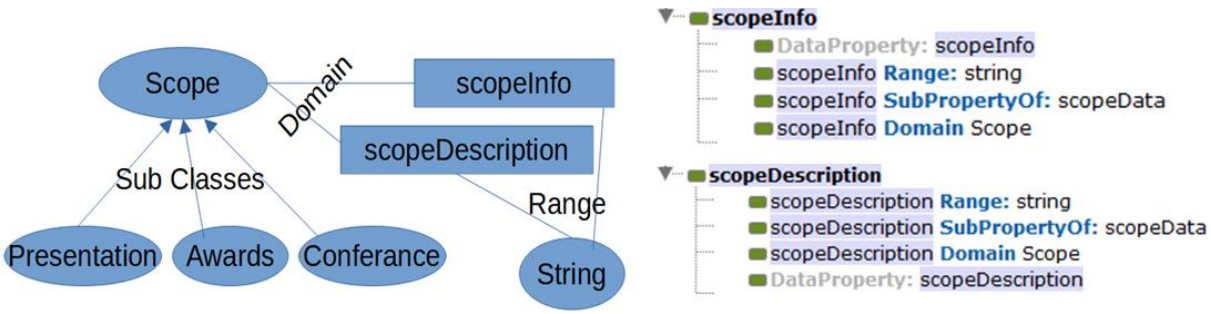
Class Happening:



Σχήμα 3.6: Διάγραμμα κλάσης Happening

Στο *Σχήμα 3.6* παρουσιάζεται η κλάση Happening. Ως κύβος παρουσιάζεται το σύνθετο πεδίο EntityAsParticipant, αν και Οντολογικά αυτό το πεδίο είναι ένα **απλό Data Property String**, το περιεχόμενο του καταχωρείται από το Πληροφοριακό Σύστημα αφού ο χρήστης έχει επιλέξει ένα Entity και ένα είδος συμμετοχής του τύπου Participant. Με το σχήμα του Εξάγωνου απεικονίζεται η διαδικασία κωδικοποίησης, όπως αναφέρεται στο παράδειγμα μετά από το *Σχήμα 3.1 Διάγραμμα πληροφοριακού συστήματος*.

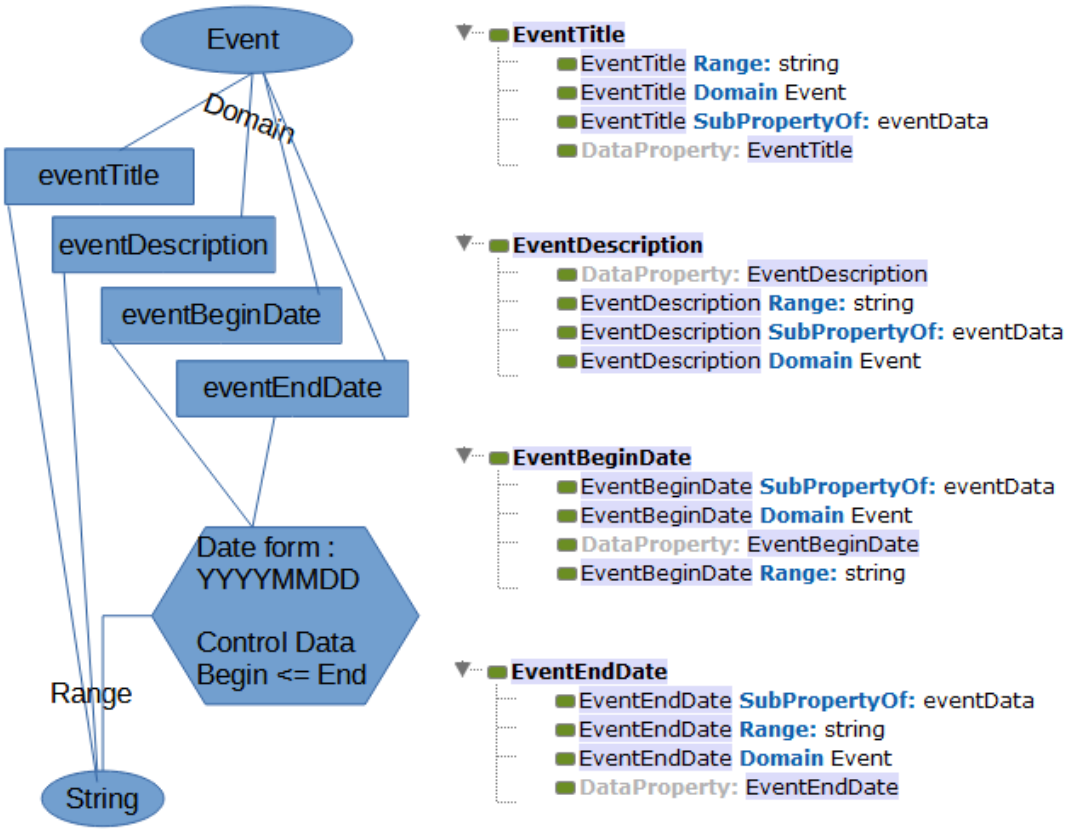
Class Scope:
Στο Σχήμα 3.7 παρουσιάζεται η δομή της κλάσης Scope με τα δεδομένα και τις σχέσεις της και δίδεται περιγραφή από το Protégé.



Σχήμα 3.7: Διάγραμμα κλάσης Scope

Σημαντικό είναι να αναφερθεί ότι: Δεν θα υπάρξουν Individual ούτε από τις υποκλάσεις της Scope, θα χρησιμοποιηθούν από το ΠΣ σαν δεύτερος τύπος κλάσης μίας Εκδήλωσης, για ομαδοποίηση των εκδηλώσεων και σε επίπεδο κλάσης, σε σχέση με τις υποκλάσεις που θα δηλώνει στο σύστημα ο χρήστης.

Class Event:



Σχήμα 3.8: Διάγραμμα κλάσης Event

Το κύριο στοιχείο της οντολογίας είναι η εκδήλωση, στο *Σχήμα 3.8* παρουσιάζεται το διάγραμμα της κλάσης **Event** με τα πεδία δεδομένων και απεικονίζεται ως **εξάγωνο** ένα σημείο ελέγχου και μορφοποίησης.

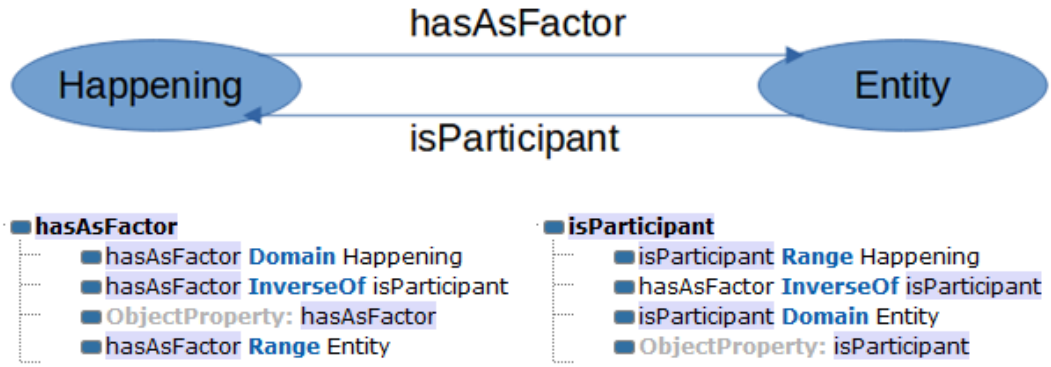
Τα πεδία **EventBeginDate** και **EventEndDate** είναι απλά String αλλά τα δεδομένα καταχωρούνται από το ΠΣ αφού έχει επιλέξει ο χρήστης ημερομηνίες από το UI, η μορφή είναι ΕΕΕΕΜΜΗΗ(Ετος – Μηνάς – Ημέρα) και γίνεται έλεγχος ώστε να προηγείται η ημερομηνία έναρξης από την λήξης.

3.1.3 Υλοποίηση Object Properties στο Protégé

Παρουσιάζονται τα Object Properties όπως φαίνονται και στο *Σχήμα 3.1 Διάγραμμα λληροφοριακού συστήματος*, χωρίς όμως να είναι ομαδοποιημένα και γίνεται μια περιγραφή της σχέσης ανά ζευγάρι πόρων.

hasAsFactor - isPatricipant

Στο Σχήμα 3.9 απεικονίζονται δύο αντίστροφες ιδιότητες αντικειμένων και το αξίωμα **InverceOf**



Σχήμα 3.9: Διάγραμμα σχέσεων Happening - Entity

Η σχέση είναι αντίστροφη και ανάλογη, αν ένα Individual κλάσης **Happening** έχει ένα Individual των υποκλάσεων της **Entity** σαν hasFactor τότε ισχύει και η αντίστροφη σχέση isParticipant. Σε περίπτωση που δηλωθεί (καταχωρηθεί) μόνο το ένα property ισχύει αυτόματα και το άλλο, αυτό γίνεται λόγω του αξιώματος **InverseOf**.

isPartOfEvent - hasHappening

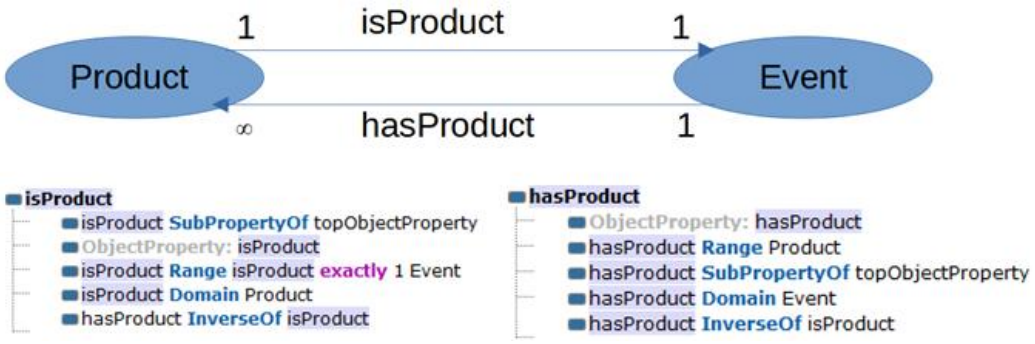
Στο Σχήμα 3.10 απεικονίζονται δύο αντίστροφες ιδιότητες αντικειμένων, το αξίωμα **InverceOf** και περιορισμοί στο πλήθος.

Σχήμα 3.10: Διάγραμμα σχέσεων Happening - Event

Το Object Property **isPartOfEvent** έχει Range **ακριβώς ένα** Individual τύπου Event, έτσι εξασφαλίζεται ότι ένα συμβάν δεν μπορεί να υπάρχει σε δύο εκδηλώσεις αλλά και το ότι ένα Individual τύπου συμβάντος θα έχει δηλωμένη εκδήλωση.

isProduct - hasProduct

Στο Σχήμα 3.11 παρουσιάζονται οι σχέσεις μεταξύ των κλάσεων Product και Event.

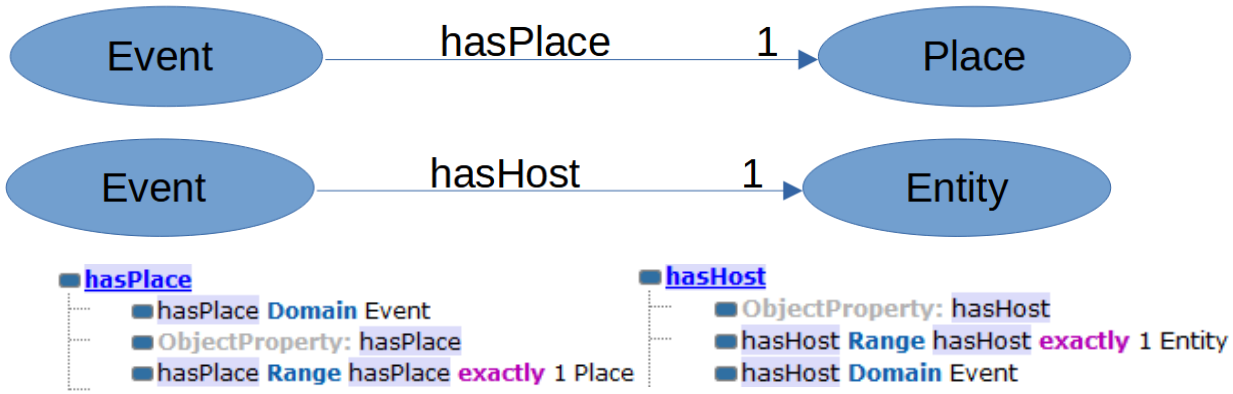


Σχήμα 3.11: Διάγραμμα σχέσεων Product - Event

Έχοντας περιορισμό στην ιδιότητα isProduct του ενός στοιχείου Event.

hasPlace - hasHost

Στο Σχήμα 3.12 παρά το γεγονός του ότι και οι δύο ιδιότητες αφορούν στο ένα άκρο τους την κλάση Event παρουσιάζονται βάσει της σχέσης τους με το άλλο άκρο, δηλαδή τις κλάσεις Place και Entity.



Σχήμα 3.12: Διάγραμμα σχέσεων Event – Place & Event – Entity

Περιορισμός υπάρχει και στις δυο ιδιότητες όσο αφορά το Range και την πληθικότητα τους.

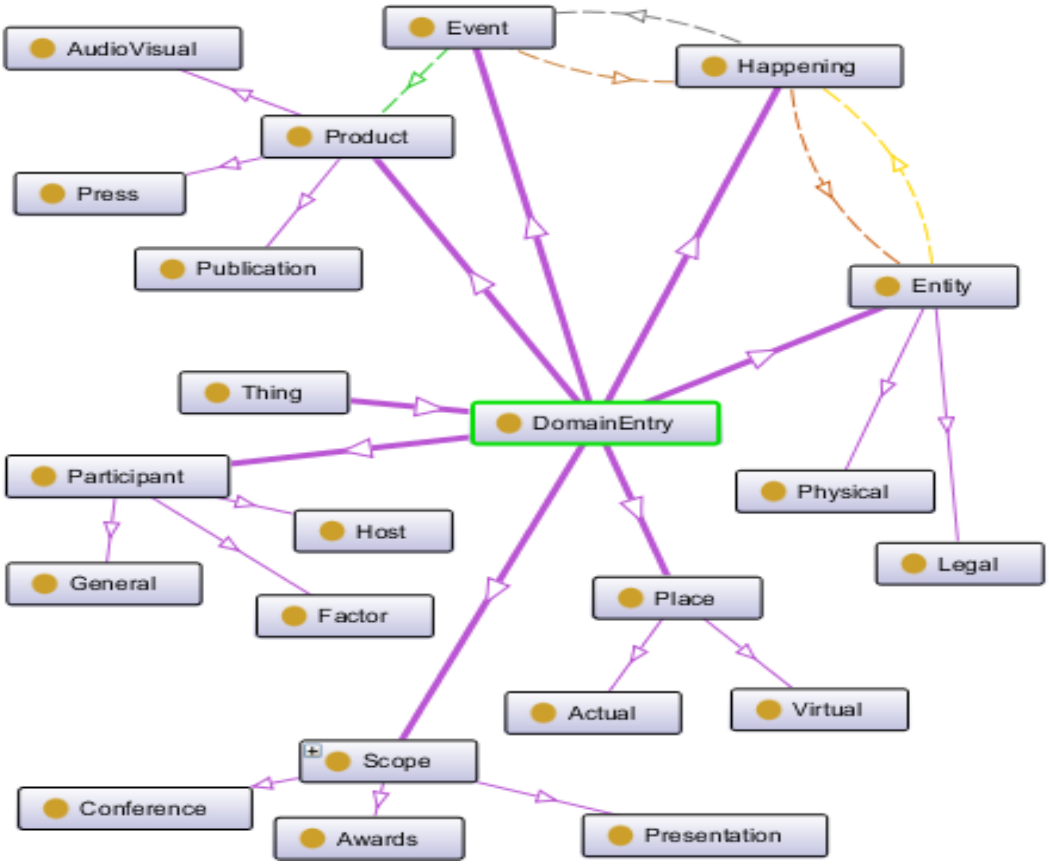
Χωρίς αυτό να αποτυπώνεται στα παραπάνω σχήματα, διευκρινίζεται ότι: **η κλάση Scope θα είναι συνώνυμη με την κλάση Event**, δηλαδή ένα Individual τύπου Event θα είναι και τύπου κάποιας υποκλάσης του Scope, αυτό βοηθά στην ομαδοποίηση των Event Individual για λήψη πληροφοριών εκδηλώσεων που ανήκουν σε κάποια υποκλάση του Scope.

3.1.4 Επισκόπηση οντολογίας

Η κενή (από Individual) οντολογία είναι η βάση εκκίνησης του πληροφοριακού συστήματος, από πλευράς δομής δεδομένων και σχέσεων, βρίσκεται στο Παράρτημα Α.

Το owl αρχείο θα ενσωματωθεί στην εφαρμογή και θα προσφέρεται σαν επιλογή για να ξεκινήσει ο χρήστης μια οντολογία από την αρχή.

Με τον Reasoner **HermiT 1.3.8** σε λειτουργία διαπιστώθηκε ότι δεν παρουσιάζεται κάποιο πρόβλημα, και το *Σχήμα 3.13* είναι το γράφημα από το Protégé που προκύπτει όταν λάβει (ανοίξει) ένα αρχείο με τον κώδικα του παραρτήματος Α.



Σχήμα 3.13: OntoGraph από Protege

3.2 Έλεγχος οντολογίας

Έχοντας υλοποιημένο το σχέδιο της οντολογίας σε αρχείο OWL το οποίο έχει της ιδιότητες, τις κλάσεις και τους περιορισμούς, θεωρήθηκε απαραίτητο να καταχωρηθούν δοκιμαστικά δεδομένα και να γίνει έλεγχος της δομής πριν την έναρξη της υλοποίησης του ΠΣ, ο έλεγχος γίνεται μέσω της καταχώρησης υλοποιήσεων (individual) κάθε κλάσης.

3.2.1 Καταχώρηση Individual

Τα Individual που καταχωρήθηκαν για όλες τις κλάσεις, για λόγους ελέγχου έχουν όνομα το οποίο περιγράφει την κλάση τους και έχουν αρίθμηση. Δηλαδή δεν αποτελούν πραγματικό δείγμα URI και αυτό έγινε για λόγους εύρεσης σφαλμάτων.

Στο Παράρτημα Β Δηλώθηκαν στοιχεία για:

- Εκδηλώσεις : 2
- Συμβάντα : 6
- Παράγωγα :4
- Τόποι :2
- Συμμετοχές : 9
- Οντότητες : 11

Η μορφή των ονομάτων είναι: «**IND-Κλάση - Υποκλάση - Αύξων Αριθμός**» με λατινικούς κεφαλαίους χαρακτήρες.

Πίνακας 3.1: Παράδειγμα ονομασίας Individual

ΟΝΟΜΑΣΙΑ	ΤΥΠΟΣ	ΚΛΑΣΗ	ΥΠΟΚΛΑΣΗ
"**IND-ENT-LEGAL-01**"	Individual	Entity	Legal
"**IND-ENT-PHY-01**"	Individual	Entity	Physical
"**IND-EVENT -01**"	Individual	Event	-
"**IND-HAPP -03**"	Individual	Happening	-
"**IND-PAR-FACTOR-02**"	Individual	Participant	Factor
"**IND-PAR-GENERAL-02**"	Individual	Participant	General
"**IND-PLACE-ACTUAL**"	Individual	Place	Actual
"**IND-PRODUCT -03**"	Individual	Product	-

Όλος ο Turtle κώδικας της δήλωσης Individual βρίσκεται στο **Παράρτημα Β** με όλες τις δηλώσεις και τα χαρακτηριστικά και για να λειτουργήσει στο Protégé πρέπει να προστεθεί (append) στον κώδικα του Παραρτήματος Α.

3.2.2 Protégé και reasoner

Παρουσιάζεται παρακάτω ενδεικτικό μέρος του κώδικα και Screen Shots από το Protégé με τον Reasoner ενεργό. Επιλέχθηκε σε κάποιες περιπτώσεις **να μην καταχωρηθούν και οι δύο αντίστροφες ιδιότητες** που αναφέρονται στο κεφάλαιο **3.1.3** ώστε να διαπιστωθεί αν ο Reasoner θα αποδώσει σωστά τις τιμές τους. Κάποιοι Individual κλάσης Entity συμμετέχουν σε συμβάντα και των **δύο εκδηλώσεων**. Τα περιεχόμενα των Data Property EntityAsParticipant, EventBeginDate και EventEndDate καταχωρήθηκαν σαν απλό String, στο Πληροφοριακό Σύστημα θα γίνεται καταχώρηση από κώδικα.

http://www.allios.dx.am#**IND-ENT-PHY-03**

:IND-ENT-PHY-03 rdf:type :Physical ,

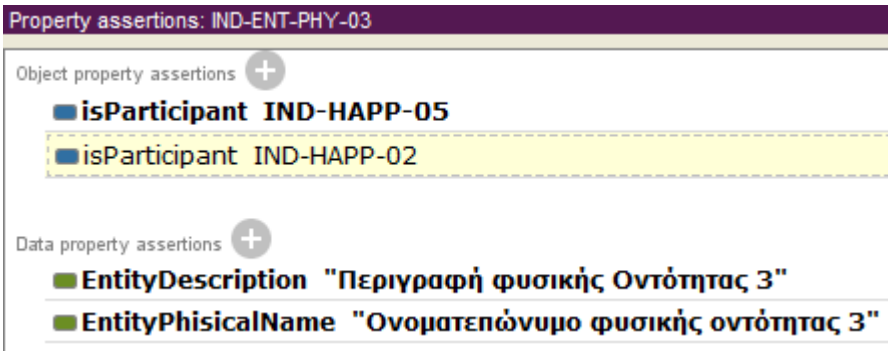
owl:NamedIndividual ;

:EntityDescription "Περιγραφή φυσικής Οντότητας 3" ;

:EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 3" ;

:isParticipant :IND-HAPP-05 .

Έχει δηλωθεί η ιδιότητα isParticipant **μόνο μία φορά** για το Happening **IND-HAPP-05**, ο Reasoner όμως, όπως φαίνεται στην Εικόνα 3.14, εξάγει το συμπέρασμα ότι ισχύει για δύο περιπτώσεις Individual τύπου Happening.



Εικόνα 3.14: Protégé IND-ENT-PY-03

Αυτό γίνεται κατανοητό βλέποντας τον κώδικα της δήλωσης του IND-HAPP-02

http://www.allios.dx.am#IND-HAPP-02

:IND-HAPP-02 rdf:type :Happening ,

owl:NamedIndividual ;

:EntityAsParticipant "IND-ENT-PHY-03 ως IND-PAR-FACTOR-01"^^xsd:string ,

"IND-ENT-PHY-05 ως IND-PAR-GENERAL-01"^^xsd:string ,

"IND-ENT-PHY-06 ως IND-PAR-GENERAL-01"^^xsd:string ;

:HappInfo "Πληροφορίες Happening 2" ;

:HappDescription "Περιγραφή Happening 2" ;

:hasAsFactor :IND-ENT-PHY-03 ,

:IND-ENT-PHY-05 ,

:IND-ENT-PHY-06 ;

:isPartOfEvent :IND-EVENT-01 .

Αφού γίνεται χρήση της αντίστροφης Ιδιότητας hasAsFactor τότε ο Reasoner συμπληρώνει σαν συμπέρασμα την Ιδιότητα isParticipant και την παρουσιάζει.

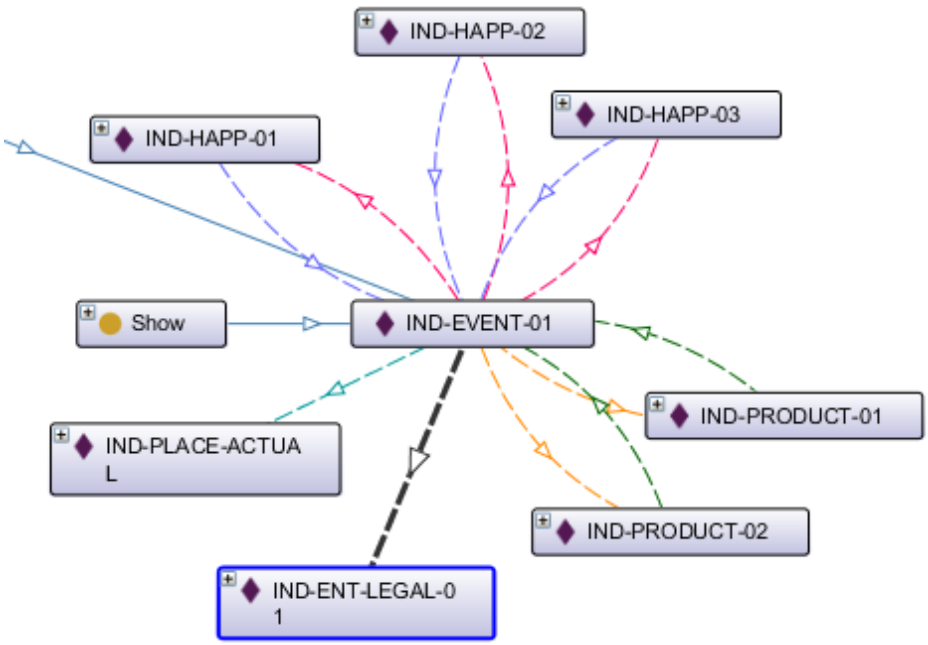
Προσοχή! *Δεν μεταβάλλεται καμία δομή, δεν προστίθεται καμία πληροφορία, δεν επηρεάζεται κανένα Μοντέλο ή αρχείο, δεν αποθηκεύεται τίποτε*.

Σαν πληροφορία του IND-HAPP-02 διαπιστώνεται ότι υπάρχει σύνδεση με μια εκδήλωση, τρείς οντότητες και τρία είδη συμμετοχής των οντοτήτων αυτών.

Παρακάτω παρουσιάζεται το σύνολο των δηλώσεων της εκδήλωσης IND-EVENT-01, με χρώμα σημειώνεται η **σύνδεση πολλών συμβάντων με μια εκδήλωση**, καθώς και η δήλωση της εκδήλωσης εκτός από κλάση Event **και σαν κλάση Show**.

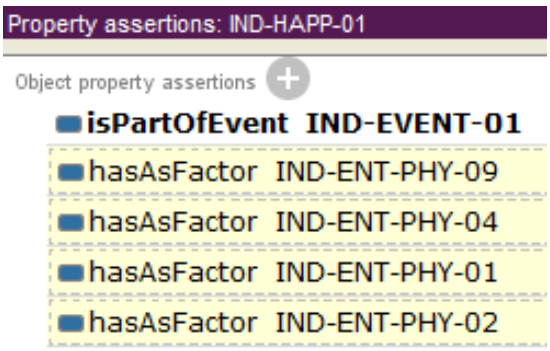
```
###  http://www.allios.dx.am#IND-EVENT-01

:IND-EVENT-01 rdf:type :Event ,  :Show ,
                owl:NamedIndividual ;
        :scopeDescription "Περιγραφή Σκοπού εκδήλωσης 1"^^xsd:string ;
        :scopeInfo "Πληροφορίες Σκοπού εκδήλωσης 1"^^xsd:string ;
        :EventDescription "Περιγραφή Εκδήλωσης 1" ;
        :EventEndDate "2021-04-01" ;
        :EventTitle "Τίτλος Εκδήλωσης 1" ;
        :EventBeginDate "2021-04-01" ;
        :hasHost :IND-ENT-LEGAL-01 ;
        :hasHappening :IND-HAPP-01 , :IND-HAPP-02 , :IND-HAPP-03 ;
        :hasPlace :IND-PLACE-ACTUAL ;
        :hasProduct :IND-PRODUCT-01 , :IND-PRODUCT-02 .
```



Σχήμα 3.15: Protégé OntoGraph IND-EVENT-01

Στο *Σχήμα 3.15* παρουσιάζεται η γραφική αναπαράσταση μέσω του tab OntoGraph στο εργαλείο Protégé, αφορά τον Individual IND-EVENT-01, μία εκδήλωση δηλαδή, με τις σχέσεις του με τα άλλα individual και αντικείμενα της δοκιμαστικής οντολογίας.



Εικόνα 3.16: Protégé IND-HAPP-01 συμπερασμός

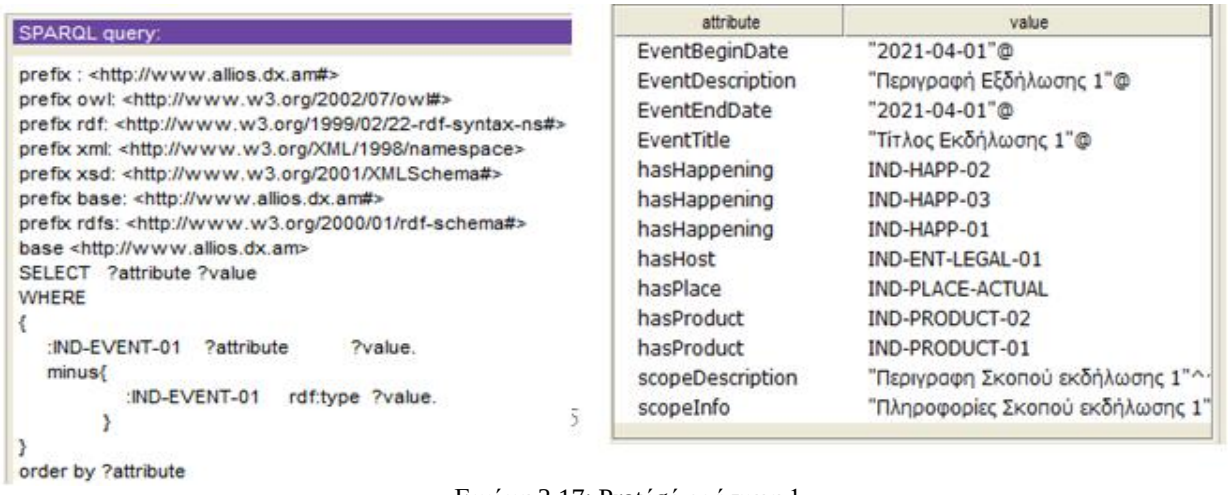
Στην *Εικόνα 3.16* παρουσιάζεται ο συμπερασμός του Reasoner για το συμβάν IND-HAPP-01 και τους Entity Individual, που προκύπτουν χωρίς να έχει γίνει δήλωση της ιδιότητας hasAsFactor σε κανένα σημείο της δήλωσης του IND-HAPP-01. Η αντίστροφη δήλωση της ιδιότητας isFactor έχει γίνει σε κάθε έναν από τους τέσσερεις Entity Individual.

3.3 SPARQL Query

Δημιουργήθηκαν SPARQL ερωτήματα με σκοπό αφ' ενός τον έλεγχο της οντολογίας έως τώρα και αφ' εταίρου να χρησιμοποιηθούν από το Πληροφοριακό Σύστημα όπως αναφέρθηκε στο 2.7.3 μέσω του μηχανισμού ARQ της βιβλιοθήκης Jena.

3.3.1 Protégé SPARQL Queries

Στο περιβάλλον του Protégé από το tab SPARQL Query δίδεται η δυνατότητα εκτέλεσης ερωτημάτων, πρέπει να δηλωθούν τα prefix εφόσον χρησιμοποιούνται. Ζητούνται **όλες οι ιδιότητες του Individual IND-EVENT-01**, (εκδήλωση του *Σχήματος 3.15* και της *Εικόνας 3.16*.)



Εικόνα 3.17: Protégé ερώτημα 1

Στην *Εικόνα 3.17 ερωτημα 1* παρουσιάζεται ο κώδικας SPARQL αριστερά και το αποτέλεσμα δεξιά, χρησιμοποιούνται δύο μεταβλητές για να εμφανιστεί η περιγραφή της ιδιότητας και η τιμή της και το MINUS χρησιμοποιείται για να μην εμφανιστούν οι ιδιότητες rdf:type που μας δίνουν Event, Show και namedIndividual. Αξίζει να σημειωθεί ότι το **IND-EVENT-01** είναι «τύπου» Event αλλά και Show αυτό έχει σαν αποτέλεσμα να χρησιμοποιούνται οι ιδιότητες του Event αλλά και του Scope,του οποίου υποκλάση είναι το Show.

Εικόνα 3.18: Protégé ερώτημα 2

Ζητούνται **όλα τα άτομα που συμμετείχαν σε κάθε εκδήλωση**, σημειώνεται ότι δεν υπάρχει άμεση συσχέτιση Ατόμου με Εκδήλωση, υπάρχει συσχέτιση Εκδήλωσης με Συμβάν και Συμβάντος με Άτομο τον κώδικα του ερωτήματος και το αποτέλεσμα παρουσιάζεται στην *Εικόνα 3.18*.

Χρησιμοποιούνται τρείς μεταβλητές: ?ΕΚΔΗΛΩΣΗ ?ΣΥΜΒΑΝ και ?ΑΤΟΜΟ, μπορούν να χρησιμοποιηθούν και Ελληνικοί χαρακτήρες αρκεί ο Η/Υ να έχει σωστά ρυθμισμένο ελληνικό locale.

Η μεταβλητή ?ΕΚΔΗΛΩΣΗ θα πάρει όλες τις τιμές των υποκειμένων που συμπληρώνει μια τριάδα με κατηγόρημα «rdf:type» και αντικείμενο «:Event».

Η μεταβλητή ?ΣΥΜΒΑΝ θα πάρει ολες τις τιμές των αντικειμένων που συμπληρώνει μια τριάδα με κατηγόρημα «:hasHappening» και υποκείμενο ?ΕΚΔΗΛΩΣΗ.

Η μεταβλητή ?ΑΤΟΜΟ θα πάρει ολες τις τιμές των υποκειμένων που συμπληρώνει μια τριάδα με κατηγόρημα «isParticipant» και αντικείμενο ?ΣΥΜΒΑΝ.

Επιλέχθηκε να μην εμφανιστεί η πληροφορία που περιέχεται στην μεταβλητή ?ΣΥΜΒΑΝ γιατί δεν ζητείται από το ερώτημα και αποτελεί ενδιάμεση πληροφορία σύνδεσης και με μια μικρή τροποποίηση του ερωτήματος της *Εικόνας 3.18*:

```
SELECT  ?ΕΚΔΗΛΩΣΗ  (count(?ΑΤΟΜΟ) as ?Σύνολο)
WHERE {
        ?ΕΚΔΗΛΩΣΗ rdf:type  :Event.
        ?ΕΚΔΗΛΩΣΗ :hasHappening ?ΣΥΜΒΑΝ.
        ?ΑΤΟΜΟ :isParticipant  ?ΣΥΜΒΑΝ
}
GROUP BY ?ΕΚΔΗΛΩΣΗ
```

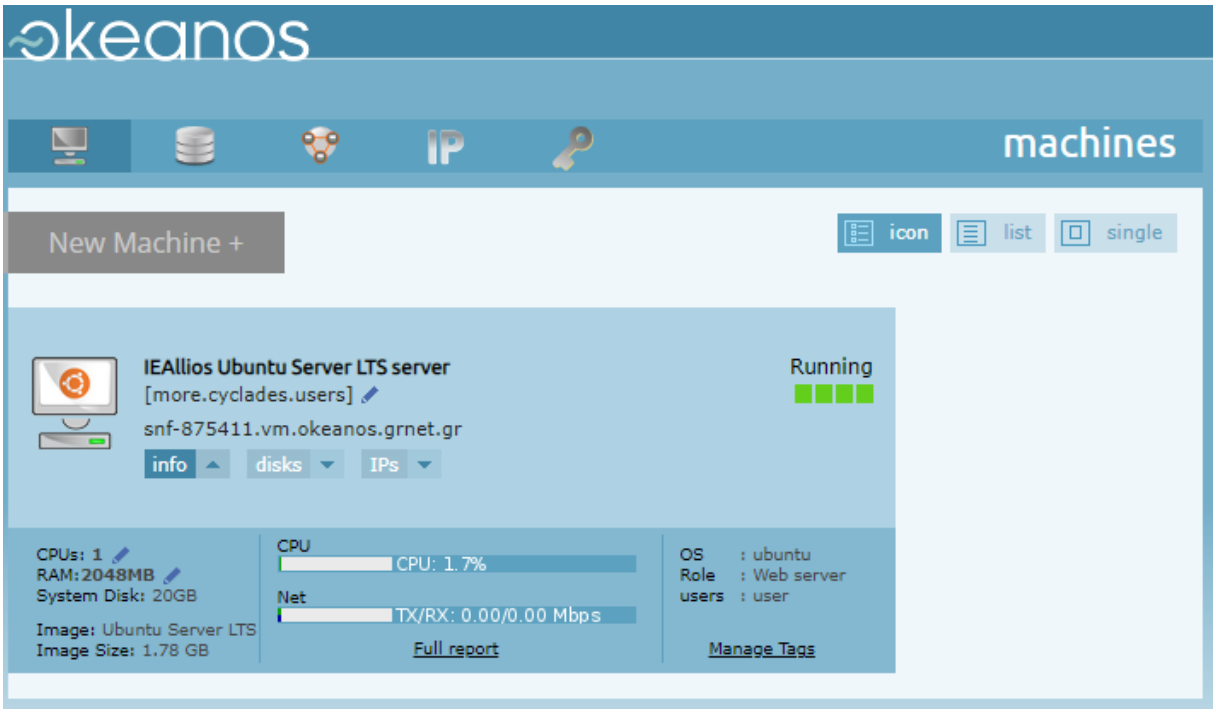
Εμφανίζεται το αποτέλεσμα στην *Εικόνα 3.19*, διαπιστώνεται ότι με την χρήση του COUNT, το σύνολο των Individual της μεταβλητής ?ΑΤΟΜΟ και ονομάζεται ?ΣΥΝΟΛΟ.

ΕΚΔΗΛΩΣΗ	Συνολο
IND-EVENT-01	"5"^^
IND-EVENT-02	"6"^^

Εικόνα 3.19: Protégé ερώτημα 3

3.3.2 Virtuoso endpoint

Στην διεύθυνση http://83.212.115.187:8890/sparql/ σηκώθηκε για τις ανάγκες της ΔΕ ένα endpoint. Είναι ένας εικονικός Server από την υπηρεσία **~Okeanos[41]**, επιλέχθηκε να γίνει εγκατάσταση του Ubuntu Server ως ΛΣ χωρίς γραφικό περιβάλλον λόγω των περιορισμών που υπάρχουν για τους δωρεάν παρεχόμενους πόρους της υπηρεσίας. Στην *Εικόνα 3.20* παρουσιάζονται συνοπτικά τα στοιχεία της εικονικής μηχανής.



Εικόνα 3.20: ~Okeaos, Ubuntu Server VM

Η εγκατάσταση του Openlink Virtuoso έγινε μέσα σε λίγα λεπτά ακολουθώντας τις οδηγίες του **VOS[42]**, αμέσως μετά την εγκατάσταση είναι διαθέσιμη πλέον η γραφική διεπαφή του Openlink Virtuoso στην οποία καταχωρήθηκαν χρήστες και ιδιότητες.

3.3.3 OpenLink Virtuoso Graph

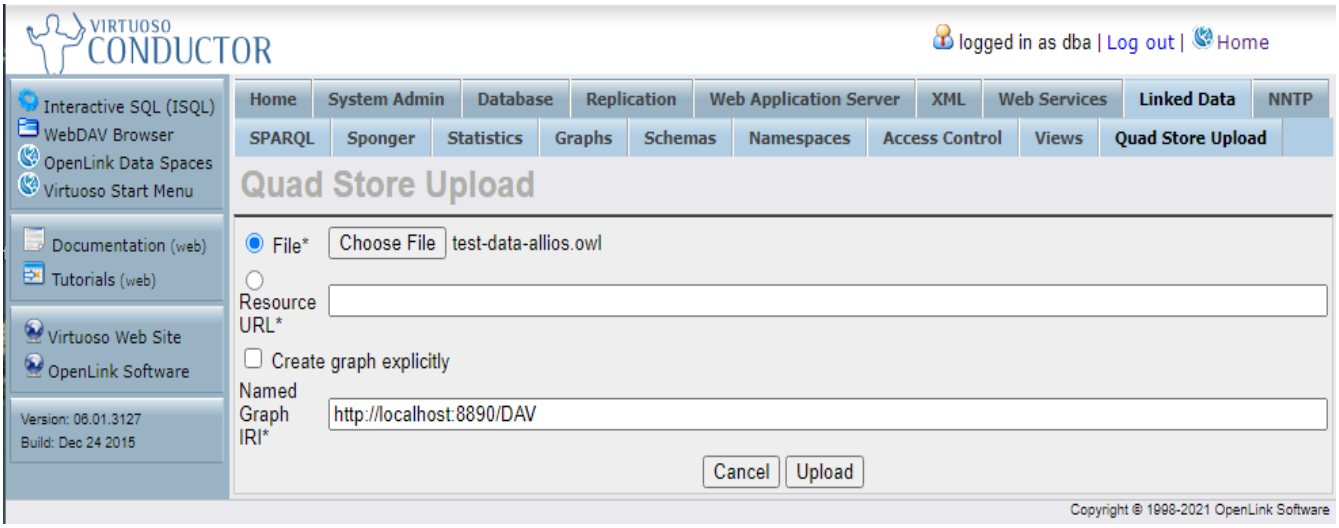
Το αρχείο της Οντολογίας που δημιουργήθηκε στο κεφάλαιο 3.1 και βρίσκεται στα Παραρτήματα Α και Β, αποθηκεύεται σε μορφή RDF για λόγους συμβατότητας με το OpenLink Virtuoso.

Ενδεικτικά παρατίθεται ο κώδικας του *Σχήματος 3.15* ως αναφορά στις διαφορές σύνταξης σε RDF.

```
<!-- http://www.allios.dx.am#IND-EVENT-01 -->
  <owl:NamedIndividual rdf:about="&base;IND-EVENT-01">
    <rdf:type rdf:resource="&base;Event"/>
    <rdf:type rdf:resource="&base;Show"/>
    <scopeDescription rdf:datatype="&xsd;string">Περιγραφη Σκοπού εκδήλωσης 1</scopeDescription>
    <scopeInfo rdf:datatype="&xsd;string">Πληροφορίες Σκοπού εκδήλωσης 1</scopeInfo>
    <EventDescription>Περιγραφή Εξδήλωσης 1</EventDescription>
```

```
        <EventEndDate>2021-04-01</EventEndDate>
        <EventTitle>Τίτλος Εκδήλωσης 1</EventTitle>
        <EventBeginDate>2021-04-01</EventBeginDate>
        <hasHost rdf:resource="&base;IND-ENT-LEGAL-01"/>
        <hasHappening rdf:resource="&base;IND-HAPP-01"/>
        <hasHappening rdf:resource="&base;IND-HAPP-02"/>
        <hasHappening rdf:resource="&base;IND-HAPP-03"/>
        <hasPlace rdf:resource="&base;IND-PLACE-ACTUAL"/>
        <hasProduct rdf:resource="&base;IND-PRODUCT-01"/>
        <hasProduct rdf:resource="&base;IND-PRODUCT-02"/>
    </owl:NamedIndividual>
```

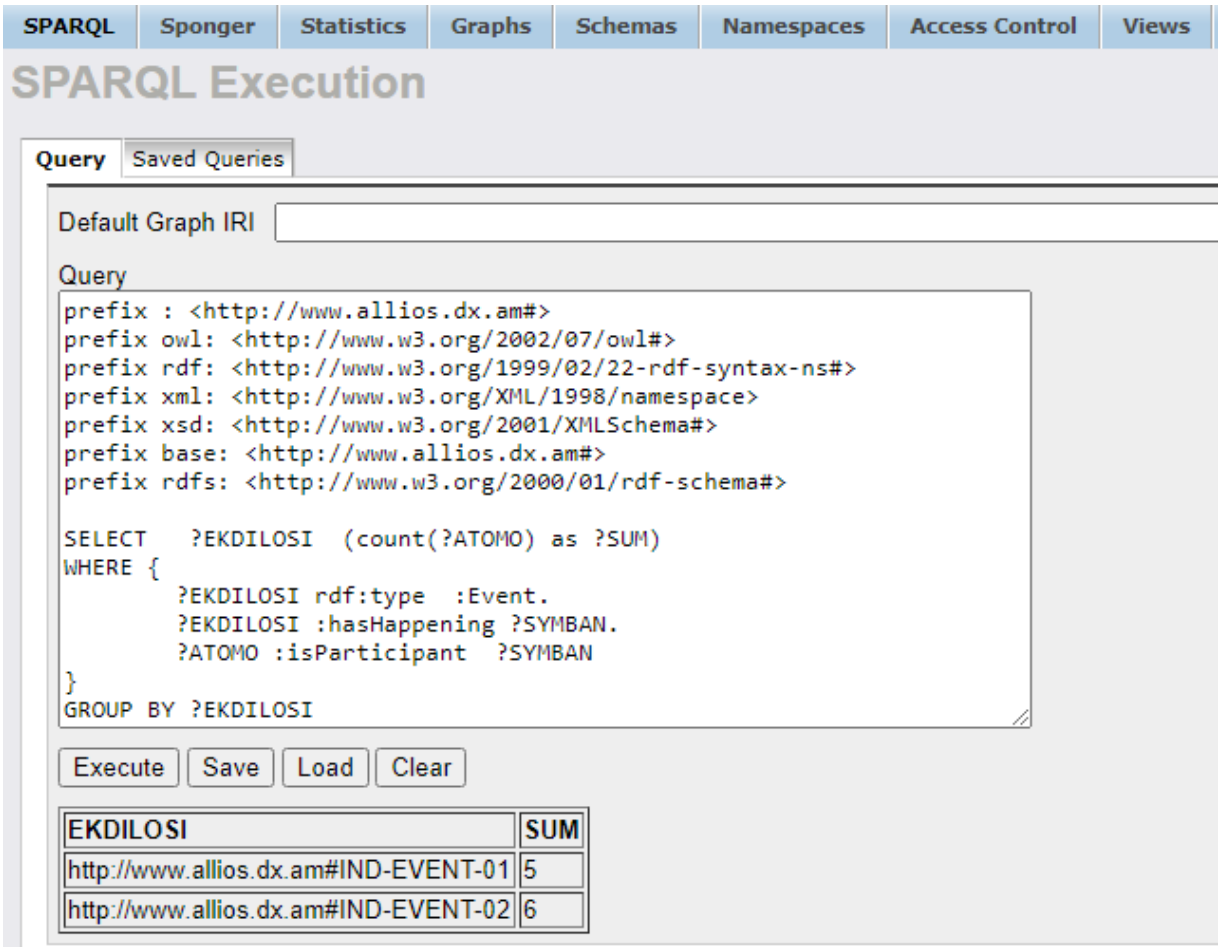
Στο **Virtuoso CONDUCTOR,** μέσω του browser, (*Εικόνα 3.21*) έχοντας κάνει login επιλέγεται το tab **Linked Data** και μετά το tab **Quad Store Upload**, με το κουμπί **Choose File** επιλέγεται το αρχείο μορφής RDF/XML που εξάχθηκε από το Protégé και γίνεται **Upload** πατώντας το κουμπί.



Εικόνα 3.21: Virtuoso Conductor Quad Store Upload

Παραμένοντας στο tab **Linked Data** επιλέγεται το tab **SPARQL** και εκτελείται το ίδιο ερώτημα που έφερε το αποτέλεσμα της *Εικόνας 3.19*, με την διαφορά του ότι δεν χρησιμοποιούνται Ελληνικοί χαρακτήρες και αφαιρέθηκε η δήλωση «base http://www.allios.dx.am».

Διαπιστώνεται στην *Εικόνα 3.22* ότι εμφανίζονται τα ίδια αποτελέσματα.

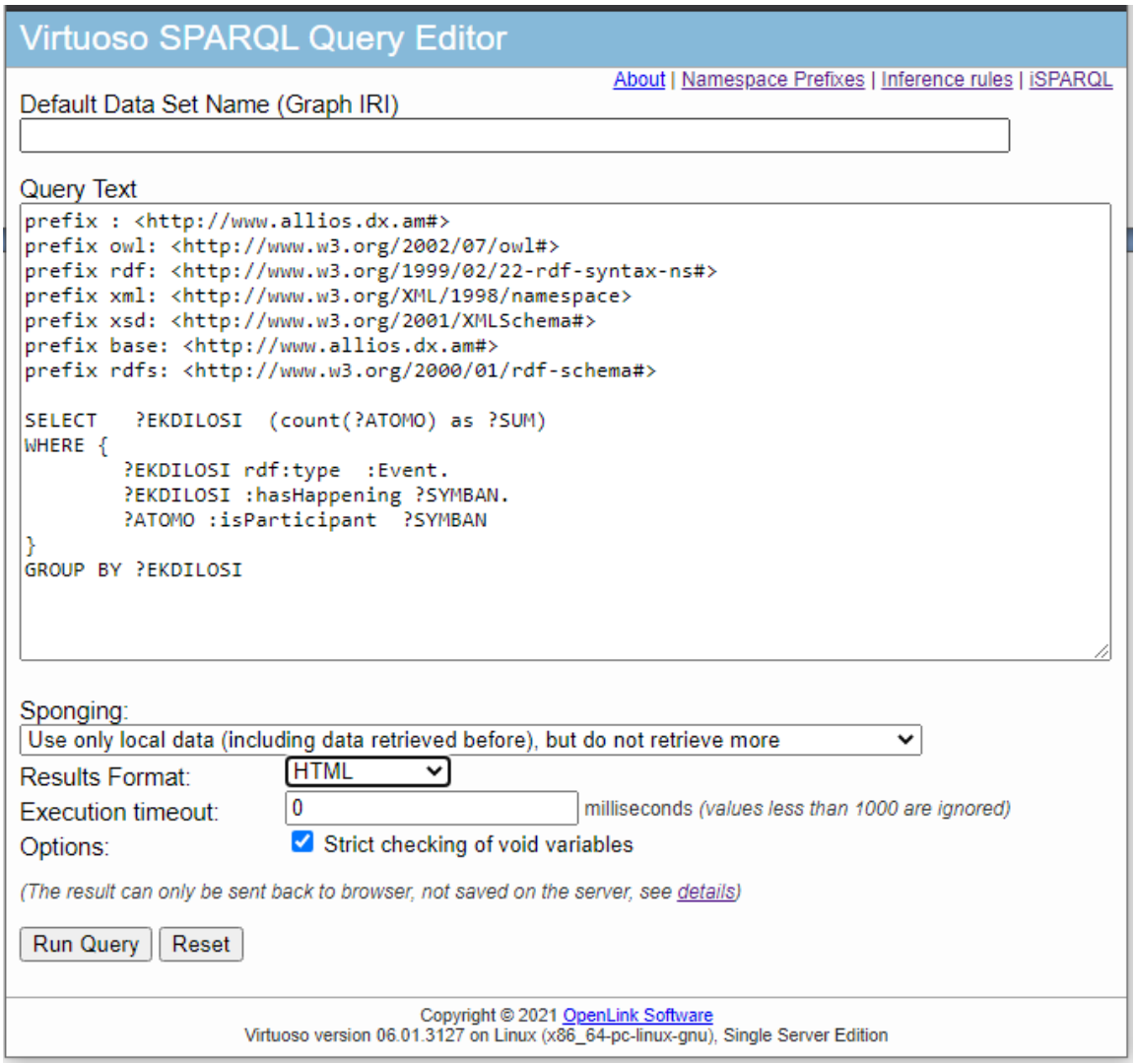


Εικόνα 3.22: Virtuoso Conductor SPARQL ερώτημα 3

Έγινε σειρά από SPARQL ερωτήματα και τα αποτελέσματα είναι **πανομοιότυπα** με τα αποτελέσματα των ερωτημάτων στο **Protégé** χωρίς προβλήματα στην χρήση Ελληνικών χαρακτήρων στις δηλώσεις (περιορισμός σε Λατινικούς χαρακτήρες υπάρχει μόνο ως προς τις μεταβλητές).

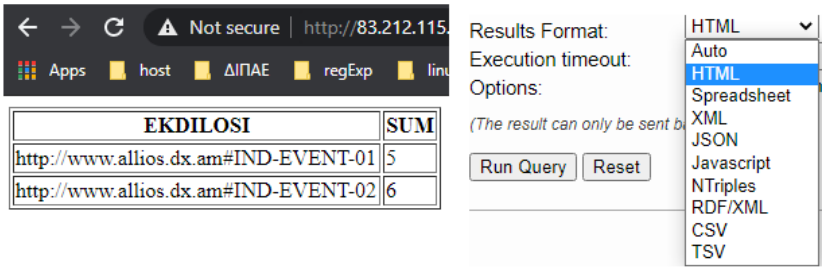
3.3.4 Δοκιμή Query

Απέναντι στο Endpoint της παραγράφου 3.3.2 γίνονται τα Queries που δοκιμάστηκαν στις παραγράφους *3.3.3 OpenLink Virtuoso Graph* και *3.3.1 Protégé SPARQL Queries*. Ο γράφος που «απαντά» στα ερωτήματα είναι το αρχείο που «ανέβηκε» στην *Εικόνα 3.21*.



Εικόνα 3.23: Virtuoso endpoint SPARQL ερώτημα 3

Η *Εικόνα 3.23* είναι στιγμιότυπο από το interface του endpoint μέσα από browser, γίνεται το ίδιο ερώτημα όπως στην *Εικόνα 3.22* και στην *Εικόνα 3.24* εμφανίζονται τα αποτελέσματα σε μορφή HTML και η επιλογή για να ληφθούν αποτελέσματα και σε άλλη μορφή μέσω της επιλογής Results Format.

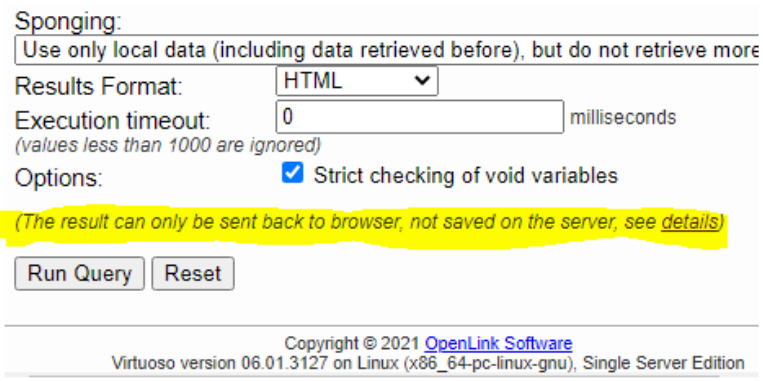


Εικόνα 3.24: Αποτέλεσμα endpoint σε μορφή HTML

Μπορεί ο χρήστης να πάρει ολόκληρη την οντολογία ή μέρος της και σε άλλα δημοφιλή format, κάνοντας το endpoint ουσιαστικά ένα εργαλείο «γέφυρα» για τα δεδομένα του γράφου και τον έξω κόσμο. Έγιναν τα ίδια ερωτήματα όπως της παραγράφου 3.3.1 και διάφορα άλλα και διαπιστώθηκε ότι εξάγονται πανομοιότυπα αποτελέσματα με τα αποτελέσματα του tab SPARQL του Protégé.

Ως συμπέρασμα θεωρήθηκε ότι η Οντολογία των Παραρτημάτων Α και Β που δημιουργήθηκε με το Protégé, συμπεριφέρεται κανονικά σε ερωτήματα SPARQL, ακόμα και σε διάφορα περιβάλλοντα.

Αξίζει να δοθεί προσοχή στην Εικόνα 3.25 που ουσιαστικά ενημερώνει ότι δεν γίνεται τροποποίηση των δεδομένων στον server.



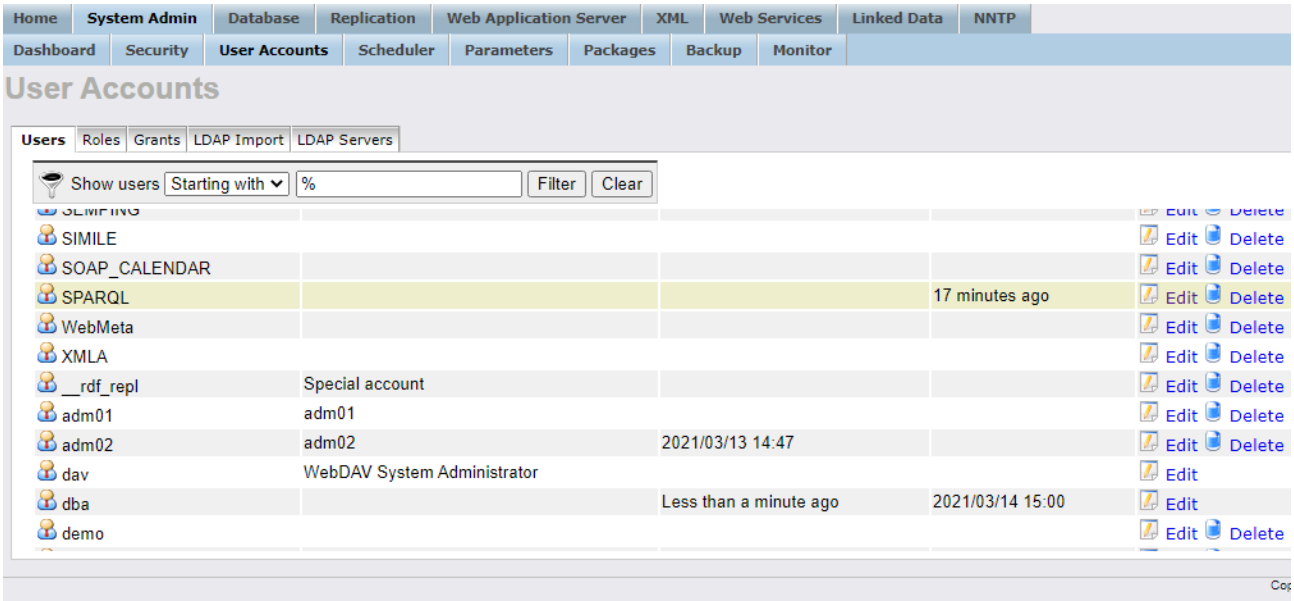
Εικόνα 3.25: Endpoint χωρίς δυνατότητα update

Το στήσιμο του endpoint υποστηρίζει μόνο ερωτήματα SELECT και SPONGE.

3.3.5 Ενεργοποίηση Update σε endpoint

Η διαδικασία της ενεργοποίησης των δυνατοτήτων πρόσβασης ενημέρωσης γίνεται σε τρία βήματα.

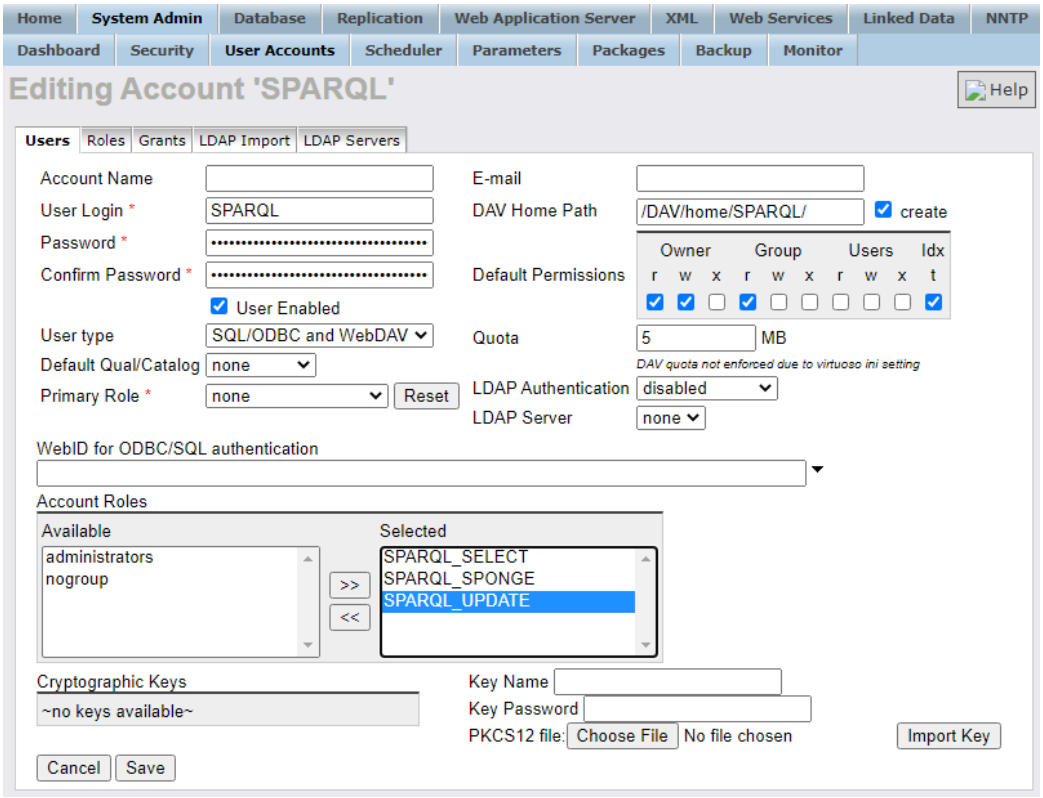
Από το tab System Admin στο Virtuoso CONDUCTOR επιλέγεται το tab User Accounts, *Εικόνα 3.26* και επιλέγεται ο λογαριασμός χρήστη, στην προκειμένη περίπτωση είναι ο User SPARQL, από εκεί μέσω της επιλογής Edit γίνεται η επεξεργασία των ιδιοτήτων αυτού του χρήστη.



Εικόνα 3.26: Conductor, User Accounts

Οι αλλαγές που πρέπει να γίνουν, παρουσιάζονται στην *Εικόνα 3.27*, είναι:

Στο **User Type**, πρέπει να είναι λογαριασμός τύπου WebDav, στην προκειμένη περίπτωση επιλέχθηκε SQL/ODBC and WebDav. Στο **Account Roles**, να επιλεγεί από τα Available το SPARQL_UPDATE και να τοποθετηθεί στο Selected. Στο **Primary Role**, επιλέγεται SPARQL_UPDATE. Στο **DAV Home Path** καταχωρείται « /DAV/home/SPARQL/ και πρέπει να **επιλεγεί το «create».**



Εικόνα 3.27: Conductor, Edit User Account

3.4 Δημιουργία εφαρμογής

Αναλύονται το περιβάλλον ανάπτυξης, ο έλεγχος εισόδου των πληροφοριών από τον χρήστη, ο τρόπος που η βιβλιοθήκη Jena αποκτά και αποδίδει την πληροφορία.

Αναλύεται η λειτουργικότητα μιας βιβλιοθήκης χρήστη με βοηθητικές συναρτήσεις που κατασκευάστηκε για το ΠΣ.

Παρουσιάζονται διαγράμματα ροής και δίνονται ενδεικτικά παραδείγματα της υλοποιημένης εφαρμογής για τις βασικές περιπτώσεις χρήσης.

3.4.1 Περιβάλλον και Βιβλιοθήκη

Για τις ανάγκες της ΔΕ επιλέχθηκε η 9^{η} έκδοση του Net Beans ως IDE ανάπτυξης, το περιβάλλον εκτέλεσης JRE του byte code είναι το 1.8.0_291 και το JDK που παράγει το byte code είναι το 1.8.0_281.

Η έκδοση της βιβλιοθήκης Jena είναι η 3.17, απαίτηση είναι να τοποθετηθεί σε σημείο που να είναι προσβάσιμο από το net beans μέσω του λειτουργικού συστήματος. Η βιβλιοθήκη πρέπει να ενσωματωθεί σαν βιβλιοθήκη χρήστη στο net beans και κατόπιν να προστεθεί στο Project.

Το αρχείο της οντολογίας σε μορφή Turtle πρέπει να βρίσκεται σε φάκελο μέσα στο Project.

3.4.2 Έλεγχος καταχώρησης δεδομένων

Ακόμα και σε ΠΣ που χρησιμοποιούν κλασικές δομές στην αποθήκευση των δεδομένων είναι απαραίτητο να γίνεται έλεγχος στην εισερχόμενη πληροφορία που δίνει ο χρήστης, μεγάλης σημασίας είναι η αποφυγή εισαγωγής χαρακτήρων οι οποίοι άμεσα ή έμμεσα μπορούν να προκαλέσουν απρόβλεπτες αντιδράσεις. Για παράδειγμα σε ένα ΠΣ βασισμένο σε script ζητείται από τον χρήστη να εισάγει το όνομα του ή την διεύθυνση του ή οτιδήποτε, θα μπορούσε να εισάγει τους κατάλληλους χαρακτήρες ώστε να προκαλέσει την εκτέλεση ενός δικού του script και να προκαλέσει από σφάλμα, στην καλύτερη περίπτωση, έως την απώλεια της ιδιοκτησίας των δεδομένων.

Πακέτα και βιβλιοθήκες ασφαλείας λύνουν αυτό το πρόβλημα σε κάθε περίπτωση, πέραν όμως τα θέματα ασφάλειας υπάρχουν και θέματα ομοιομορφίας και σωστής δομής της πληροφορίας.

Στην περίπτωση μας επιλέχθηκε η αποθήκευση της πληροφορίας σαν οντολογία με σύνταξη turtle και η SPARQL σαν γλώσσα εκτέλεσης αποθήκευσης, ενημέρωσης και διαγραφής και αποφασίστηκε να χρησιμοποιηθεί και το ίδιο το predicate σαν πληροφορία, για να έχουμε οντολογικά “αναγνωρίσιμο” αποτέλεσμα.

Θα μπορούσε ο χρήστης να εισάγει, καλόβουλα, αντί του μικρού ονόματος “Uwe”, το “U;e” δηλαδή το αντί του πλήκτρου “W” να πατήσει το “Q” στα Ελληνικά ή ακόμα (πιο πιθανό) αντί του επωνύμου “Boll” να περάσει “Bo;;” ή “Bo;l” ή “Bol;” που σε αυτή την περίπτωση το «;» είναι ακριβώς δίπλα από το “L”. Τα προβλήματα ξεκινάν από τις τριάδες, η κάθε πληροφορία θα αποθηκευθεί σαν τριάδα στο σύστημα, κάθε τριάδα τελειώνει με μια τελεία, αν τελειώνει με «;» σημαίνει ότι η επόμενη τριάδα έχει το ίδιο πρώτο μέρος (predicate) με την τρέχουσα.

Το αποτέλεσμα θα είναι να μην γίνει (αν γίνει) σωστή αποθήκευση.

Αν το «;» με κάποιο τρόπο περάσει στην οντολογία θα προκαλέσει απρόβλεπτες συμπεριφορές στα queries, που τυχών θα τρέξουν στο μέλλον, αφού θα έχει διαταράξει την ονοματολογία και τις σχέσεις της οντολογίας.

Σχεδιάστηκε το ΠΣ με τέτοιο τρόπο έτσι ώστε να αποτρέπεται η εισαγωγή των οντολογικά ευαίσθητων χαρακτήρων κατά την στιγμή της πληκτρολόγησης, ώστε ο χρήστης να έχει ενημέρωση, να βλέπει δηλαδή, την πληροφορία που τελικά θα αποθηκευτεί.

Αφού επιλέχθηκε να δίνεται η δυνατότητα στον χρήστη να ορίσει τα ονόματα των Individual αλλά και ονόματα μερικών Sub Class, αποφασίσθηκε να εφαρμοστούν επιπλέον κανόνες κατά την πληκτρολόγηση.

Αποφάσεις περί αυτού πάρθηκαν αλλά έγινε η σχεδίαση με τρόπο τέτοιον ώστε να μπορούν να μεταβληθούν οι κανόνες σχετικά εύκολα από τον προγραμματιστή.

Συγκεκριμένα:

Ένα Class ή ένα Individual πρέπει να ξεκινά οπωσδήποτε με χαρακτήρα (όχι σύμβολο ή αριθμό) και μάλιστα κεφαλαίο στην περίπτωση του Class

Ένα Class ή ένα Individual πρέπει να μην έχει όνομα ίδιο με τα άλλα.

Οι οντολογίες όχι απλά το επιτρέπουν αυτό αλλά **βασίζονται σε αυτό,** υπάρχει η λογικής της πολύπλευρης σημασίας των πραγμάτων, δηλαδή:

Δεν υπάρχει πρόβλημα αν ισχύουν τα

“Γιάννης-Παπαδόπουλος είναι Άνθρωπος”,

“Γιάννης-Παπαδόπουλος είναι Καλεσμένος”,

“Γιάννης-Παπαδόπουλος είναι Υπολοχαγός”,

Τότε ο ισχυρισμός ότι “Υπολοχαγός είναι Καλεσμένος” είναι σωστός γιατί προκύπτει βάσει συλλογισμού και εφόσον δεν απορρίπτεται από άλλες συνθήκες.

Καταχωρώντας έναν ισχυρισμό τύπου “*Γιάννης-Παπαδόπουλος είναι...*” θα γίνει φυσικά δεκτός.

Καταχωρώντας ένα νέο Individual ενημερώνεται ο χρήστης, **την στιγμή της πληκτρολόγησης** ότι υπάρχει Individual με αυτό το όνομα και αν αγνοήσει την ειδοποίηση, και συνεχίσει χωρίς να το αλλάξει, τότε καταχωρείται το όνομα που έδωσε ο χρήστης αλλά ακολουθούμενο από το String “A-COPY” εξασφαλίζοντας έτσι την μοναδικότητα του individual προγραμματιστικά.

Επιλέχθηκε να μην γίνονται δεκτά στο όνομα Class και Individual τα «κενά», και δημιουργήθηκαν τρείς συναρτήσεις τροποποίησης με τα εξής αποτελέσματα:

Σε ενδεχόμενη είσοδο του χρήστη:

“Αυτό είναι μια κλάση “ ή

“ ΑΥΤΟ ΕΙΝΑΙ ΜΙΑ ΚΛΆΣΗ” ή

“ΑυτΟ εΙΝΑΙ μΙα ΚλΆΣΗ”ή

“Αυτ:ο εί;ναι μ^ΙΑ κλά@σ#η”

θα καταχωρηθεί εγγυημένα ένα από τα τρία παρακάτω:

“*ΑυτοΕίναιΜιαΚλάση*” ή ”*Αυτό_ειναι_μία_κλάση*” ή ”*Αυτό-είναι-μια-κλάση*”

ανάλογα με την συνάρτηση ελέγχου που θα επιλεχθεί για να υπάρχει ομοιομορφία στα δεδομένα.

3.4.3 Jena και πρόσβαση στη πληροφορία

Σε ένα ΠΣ η πρόσβαση στην πληροφορία έχει ρόλο που δεν αφορά τον χρήστη, θα μπορούσαν να υλοποιηθούν δεκάδες διαθέσιμες τεχνολογίες και δομές για να παρέχουν αυτή την πρόσβαση, δεν θα έπρεπε να επηρεάζει ούτε την ανάπτυξη μιας εφαρμογής ο τρόπος πρόσβασης στην πληροφορία και μάλιστα αυτός ο τρόπος επιλέγεται κατά την διάρκεια της σχεδίασης του ΠΣ.

Στην περίπτωση αυτής της έρευνας είναι δεδομένη η μορφή της πληροφορίας, Οντολογίες, στο κεφάλαιο 2 έγινε η βασική σχεδίαση του ΠΣ με όσο πιο γενικό τρόπο γίνεται χωρίς να επηρεαστεί από την δεδομένη μορφή της πληροφορίας, συνεχίστηκε η ανάπτυξη υπό την ιδία λογική και χρησιμοποιήθηκαν από την βιβλιοθήκη Jena τα στοιχεία εκείνα που θα περίμενε να βρει, ένας προγραμματιστής, σε διάφορες «δημοφιλείς» δομές πληροφορίας.

Το Interface **Model[40]** της Jena επεκτείνει τα μοντέλα Lock, ModelCon, ModelGraphInterface, PrefixMapping, RDFReader, RDFWriter, και εξειδικεύεται από τα **OntModel** και **InfModel**.

Με μια γενικότητα και απλοϊκότητα στην έκφραση, ορίζεται ότι ένα μοντέλο είναι η «βάση δεδομένων» ή «ο κατάλογος δεδομένων» ή το «αρχείο» με τα δεδομένα… γενικά τα Δεδομένα.

Προσφέρει στον προγραμματιστή εγγραφή και ανάγνωση από το υλικό, δημιουργείται δηλαδή ένα αντικείμενο Model στην μνήμη από ένα αρχείο του ΛΣ το οποίο περιέχει τις οντολογικές πληροφορίες, αλλά δημιουργεί και ένα αρχείο στο ΛΣ από ένα αντικείμενο Model που υπάρχει στην μνήμη, αυτό γίνεται με το **ModelFactory** που παρέχει μεθόδους για την δημιουργία τους.

Model.createOntologyModel

Επιστρέφει ένα αντικείμενο Model σύμφωνα με τις παραμέτρους, στην εφαρμογή χρησιμοποιήθηκε η παράμετρος OntModelSpec.OWL_DL_MEM που ορίζει ένα μοντέλο τύπου OWL DL στην μνήμη.

Model.write

Έχει σαν έξοδο το Μοντέλο με μια συγκεκριμένη σύνταξη, σύμφωνα με τις παραμέτρους. Στην εφαρμογή χρησιμοποιήθηκε ως πρώτη παράμετρος ένα αντικείμενο τύπου **FileWriter[43]** που ορίζει ένα αρχείο που θα γραφτεί στον δίσκο με τις αντίστοιχες ιδιότητες του, και ως δεύτερη παράμετρος το String "TURTLE" ή το String "RDF/XML-ABBREV" ανάλογα με την επιθυμητή σύνταξη που επιλέχθηκε να αποθηκευτεί το αρχείο.

Model.close

Κλείνει το Μοντέλο και απελευθερώνει τους πόρους που κατείχε.

Το Αντικείμενο **Query[44]** της Jenaεπεκτείνει αντικείμενο **Prologue** (μέρος του sparql.core). Είναι μια δομή δεδομένων για τις ανάγκες ενός ερωτήματος και παρέχει μια σειρά μεθόδων που ανταποκρίνονται τόσο στην **Οντολογική** διάσταση όσο και στην κλασική **SQL** λειτουργικότητα που περιμένει ένας προγραμματιστής. Παρέχει και το **Clonable** και το **Printable** Interface, και για δημιουργία Query χρησιμοποιήθηκε το **QueryFactory** με τις μεθόδους που παρέχει.

QueryFactory.create

Επιστρέφει μία δομή τύπου **Query** σύμφωνα με το String που δέχεται σαν παράμετρο, το String αυτό είναι ουσιαστικά το SPARQL select που ζητείται. Πρέπει πάντα να περικλείεται από try/catch block γιατί σε περίπτωση λάθος «πετά» **QueryException**.

QueryFactory.read

Επιστρέφει μία δομή τύπου **Query** σύμφωνα με το String που δέχεται σαν παράμετρο, το String αυτό είναι ουσιαστικά μια διαδρομή προς ένα αρχείο και το αρχείο αυτό περιέχει το SPARQL select που ζητείται. Πρέπει πάντα να περικλείεται από try/catch block γιατί σε περίπτωση λάθος «πετά» **QueryException**.

Το Interface **QueryExcecution[45]** της Jena επεκτείνει τα μοντέλα QueryEngineHTTP και QueryExecutionBase, και εκτελεί ένα Query για μία φορά, χρησιμοποιήθηκε το QueryExcecutionFactory για την δημιουργία του με τις μεθόδους που παρέχει.

QueryExcecutionFactory.create

Έχει σαν έξοδο το **QueryExecution** που είναι το ερώτημα, που τέθηκε σαν πρώτη παράμετρος, πάνω στο μοντέλο, που τέθηκε σαν δεύτερη παράμετρος.

QueryExcecutionFactory.sparqlService

Έχει σαν έξοδο το **QueryExecution** που εκτελέστηκε μέσω HTTP.

Το Interface **ResultSet[46]** της Jena επεκτείνει τα μοντέλο Iterator<QuerySolution>, παρέχει σε μορφή πίνακα τα αποτελέσματα ενός Select, η κάθε γραμμή είναι ένα σετ με μεταβλητές και περιεχόμενα που ικανοποιούν την συνθήκη του ερωτήματος.

ResultSet.getResourceModel

Επιστρέφει ένα αντικείμενο τύπου **Model** βάσει του οποίου δημιουργήθηκαν τα αποτελέσματα.

ResultSet.getResultVars

Επιστρέφει ένα **String List** που περιέχει τα ονόματα των μεταβλητών (στηλών) του πίνακα των αποτελεσμάτων, για να χρησιμοποιηθούν για την εμφάνιση των τιμών της τρέχουσας γραμμής αποτελεσμάτων.

ResultSet.getRowNumber

Επιστρέφει ένα **int** που είναι ο αριθμός της τρέχουσας γραμμής αποτελεσμάτων.

ResultSet.hasNext

Επιστρέφει **true/false** στην ερώτηση αν υπάρχει επόμενο αποτέλεσμα.

ResultSet.next

Επιστρέφει το επόμενο αντικείμενο **QuerySolution** από στο σετ των λύσεων.

Το Interface **QuerySolution[47]** της Jena δίνει την λειτουργικότητα που χρειάζεται για την προσπέλαση στην πληροφορία μιας απάντησης ενός ερωτήματος.

QuerySolution.contains

Επιστρέφει **true/false** αν υπάρχει η μεταβλητή της παραμέτρου.

QuerySolution.get

Επιστρέφει ένα αντικείμενο τύπου **RDF/NODE** που αντιστοιχεί στην μεταβλητή της παραμέτρου.

QuerySolution.getResource

Επιστρέφει ένα αντικείμενο τύπου **Resource** που αντιστοιχεί στην μεταβλητή της παραμέτρου.

QuerySolution.varNames

Επιστρέφει ένα αντικείμενο τύπου **String Iterator** για να χρησιμοποιηθεί σε βρόγχο (τύπου for each) για την προσπέλαση των ονομάτων των μεταβλητών.

Η εκτέλεση **Update Query** για την προσθήκη τριάδων στο ΠΣ αλλά και για την αλλαγή τους με Delete Insert γίνεται μέσω της Κλάσης **UpdateAction[48]** και της συνάρτησης **parseExecute**.

UpdateAction. parseExecute

Εκτελεί το **ερώτημα** που βρίσκεται στην πρώτη String παράμετρο, στο **Μοντέλο** που βρίσκεται στην δεύτερη παράμετρο.

3.4.4 Περιγραφή βοηθητικών συναρτήσεων

Με τον γράφο **(**Παράρτημα Α και Β) δημιουργήθηκαν συναρτήσεις κάνοντας χρήση των εργαλείων της βιβλιοθήκης Jena, ο σκοπός είναι η επαναχρησιμοποίηση τους για την αποφυγή μεγάλου και δυσνόητου κώδικα Java, κώδικας των συναρτήσεων βρίσκεται στο Παράρτημα C**.**

MySparqlAuth

Είναι μια Java Class που όταν χρειάζεται η εφαρμογή να στείλει ερώτημα σε endpoint το οποίο έχει Authentication με όνομα χρήστη και κωδικό, πρέπει η εφαρμογή να είναι μια υλοποίηση της κλάσης αυτής.

Say(String string)

Ένας σύντομος τρόπος του System.out.println().

getMyPrefix()

Είναι μια συνάρτηση χωρίς παραμέτρους που επιστρέφει στο όνομα της ένα String που είναι το Prefix της εν λόγω Οντολογίας.

getMyPrefix(String url)

Η ίδια συνάρτηση με παράμετρο ένα URL μιας οντολογίας σε περίπτωση που υπάρχει αλλαγή στην ονομασία.

clippHTTP(String uri)

Επιστρέφει το όνομα ενός URI πόρου χωρίς το URL μέρος, π.χ. ένα URI στην *Εικόνα 3.24* είναι το ” http://www.allios.dx.am#IND-EVENT-01,” θα επιστραφεί “IND-EVENT-01.”.

noNewLinesNoTabs(String theString)

Επιστρέφει το εισερχόμενο String χωρίς αλλαγή γραμμής και tab αντικαθιστώντας τα με ένα κενό το κάθε ένα, αφορά τον έλεγχο των στοιχείων που εισάγει ο χρήστης.

specialCharRemove(String theString)

Επιστρέφει το εισερχόμενο String απαλλαγμένο από «ειδικούς» χαρακτήρες. Ουσιαστικά επιλέχθηκε η απόρριψη από το κείμενο που καταχωρεί ο χρήστης κάποιων χαρακτήρων που θα μπορούσαν να προκαλέσουν απρόβλεπτη συμπεριφορά του ΠΣ.

Οι χαρακτήρες αυτοί είναι:

\t, \n, semicolon, coma, dot, ^, ’, ”, #, /, \, <, >, @

makeLegal(String theString)

Επιστρέφει το εισερχόμενο String απαλλαγμένο από «ειδικούς» Σημασιολογικά χαρακτήρες. Σχεδιάστηκε για την αποφυγή εισαγωγής αυτών των χαρακτήρων σε ονόματα URI που καταχωρεί ο χρήστης, π.χ. ζητείται να καταχωρήσει το όνομα ενός νέου Individual Scope, αν στο text πεδίο του UI του ΠΣ καταχωρήσει «Philanthropy» δεν υπάρχει κανένα πρόβλημα, αλλά αν καταχωρήσει «Phi;anthropy» ; (το «L» και το «;» είναι το ένα διπλά στο άλλο) και αυτό περάσει στο SPARQL Update που ετοιμάζει το ΠΣ, τότε στην καλύτερη περίπτωση δεν θα εκτελεστεί το Update, αφού το «;» έχει ειδική σημασία για μια τριάδα στην Turtle σύνταξη.

Οι χαρακτήρες αυτοί είναι:

\t, \n, semicolon, coma, dot, ^, ’, ” και :

makeLegalClass(String theString)

Το ίδιο με την makeLegal αλλά με τροποποιημένη την λίστα χαρακτήρων σε σχέση με μια κλάση.

getModelFromTTLFile()

Επιστρέφει ένα αντικείμενο Model το οποίο δημιουργείται από το αρχείο που περιχέει την βασική και embedded οντολογία του ΠΣ.

getModelFromTTLFile(String theFile)

Επιστρέφει ένα αντικείμενο Model το οποίο δημιουργείται από το αρχείο που ορίζεται στο εισερχόμενο String.

writeModelToTurtle(Model theModel, String theFile)

Αποθηκεύει το εισερχόμενο Μοντέλο με σύνταξη Turtle στο αρχείο που δίνει ο χρήστης. Πρέπει να περικλείεται από try/catch block λόγω της χρήσης του FileWriter.

WriteModelToRDF(Model theModel, String theFile)

Αποθηκεύει το εισερχόμενο Μοντέλο με σύνταξη RDF/XML στο αρχείο που δίνει ο χρήστης. Πρέπει να περικλείεται από try/catch block λόγω της χρήσης του FileWriter.

getAllUserNames(Model theModel)

Από το εισερχόμενο Μοντέλο, επιστρέφει σε μορφή ArrayList όλα τα URI στοιχεία της Οντολογίας που έχουν καταχωρηθεί από τον χρήστη του ΠΣ, ο σκοπός της συνάρτησης αυτής είναι να παρέχει στον προγραμματιστή μια λίστα που περιέχονται όλα τα ονόματα που καταχώρησε ο χρήσης σε Class, DatatypeProperty και ObjectProperty ώστε να τον εμποδίσει να τα ξαναπεράσει.

getValuesOfPropertiresOfIndividual(Model theModel, String individual)

Από το εισερχόμενο Μοντέλο και το εισερχόμενο όνομα του Individual, επιστρέφει σε μορφή ArrayList όλες τις τιμές (δεδομένα) ιδιοτήτων που έχει το Μοντέλο για τον Individual. Ο σκοπός είναι να δώσει στον προγραμματιστή έναν γρήγορο τρόποι να προσπελάσει τα στοιχεία αυτά χωρίς την χρήση κόμβων και χωρίς να χρειάζεται γνωρίζει τα ονόματα των ιδιοτήτων.

getValuesOfPropertiresOfClass(Model theModel, String theClass)

Από το εισερχόμενο Μοντέλο και το εισερχόμενο όνομα της κλάσης, επιστρέφει σε μορφή ArrayList όλες τις τιμές (δεδομένα) ιδιοτήτων που έχει το Μοντέλο για την Κλάση. Ο σκοπός είναι να δώσει στον προγραμματιστή έναν γρήγορο τρόποι να προσπελάσει τα στοιχεία αυτά χωρίς την χρήση κόμβων και χωρίς να χρειάζεται γνωρίζει τα ονόματα των ιδιοτήτων.

getMembersOfClassInArrayList(Model theModel, String className)

Από το εισερχόμενο Μοντέλο και το εισερχόμενο όνομα της κλάσης, επιστρέφει σε μορφή ArrayList όλους τους Individual που αποτελούν υλοποιήσεις της κλάσης.

getMembersOfClass(Model theModel, String className)

Είναι ακριβώς ίδια με την getAllIndividualsOfClass(Model theModel, String className) με την διαφορά στο ότι επιστρέφει αντικείμενο τύπου ObservableList το οποίο είναι ευκολότερα διαχειρίσιμο από την JavaFX.

getMembersOfClass(String theFile, String className)

Είναι ακριβώς ίδια με την getMembersOfClass με την διαφορά στο ότι η πληροφορία αναζητείται όχι στο Μοντέλο αλλά σε αρχείο στον δίσκο.

getAllIndividualsOfSuperClass(Model theModel, String className)

Από το εισερχόμενο Μοντέλο επιστρέφει σε μορφή ArrayList όλους τους Individual που αποτελούν υλοποιήσεις των υποκλάσεων της εισερχόμενης κλάσης. Αφού επιλέχθηκε να δίνεται η δυνατότητα στον χρήστη του ΠΣ να δημιουργεί υποκλάσεις, με αυτήν την συνάρτηση ο προγραμματιστής λαμβάνει τους Individual χωρίς να χρειάζεται γνωρίζει τα ονόματα των κλάσεων που καταχώρησε ο χρήστης.

getObervableLisFromQueryResults(ResultSet results)

Δέχεται σαν είσοδο ένα αντικείμενο τύπου ResultSet, διαβάζει για κάθε σετ αποτελεσμάτων τα δεδομένα και φτιάχνει μια ObservableList, τα δεδομένα απαλλάσσονται από τα

σημασιολογικά τους χαρακτηριστικά και είναι έτοιμα να χρησιμοποιηθούν για την δημιουργία μενού, λιστών και άλλων γραφικών στοιχείων στην JavaFX.

getArrayLisFromQueryResults(ResultSet results)

Εχει ακριβώς την ίδια λειτουργία με την getObervableLisFromQueryResults με την διαφορά στο ότι επιστρέφει ArrayList.

updateQueryOnTTLFile(String myFile, String queryString)

Δέχεται σαν είσοδο την διαδρομή του αρχείου και το όνομα σε μορφή String, ένα Query Update String. Εκτελεί σε ένα προσωρινό Μοντέλο το Update και μετά ενημερώνει το αρχείο (γράφει το Μοντέλο στον δίσκο).

renameAll(String theFile, String oldName, String newName)

Δέχεται σαν είσοδο την διαδρομή του αρχείου και το όνομα σε μορφή String, το «παλιό» όνομα του URI που θα αλλάξει και το «νέο» όνομα που θα έχει. Γίνεται ενημέρωση του αρχείου κάνοντας ουσιαστικά ένα DELETE, INSERT, WHERE, για όλες τις τριάδες που έχουν σαν Κατηγόρημα το «παλιό» όνομα, σε όλη την οντολογία του αρχείου.

getFrontFromMixString(String mixedTextField)

Δέχεται ένα String του οποίο το περιεχόμενο είναι η τιμή της Ιδιότητας Δεδομένων με το όνομα EntityAsParticipant του Σχήματος 3.6 και για να υλοποιηθεί η λειτουργία που περιγράφεται για το πεδίο αυτό στην 3.1.1, επιστρέφει το αριστερό μέρος του EntityAsParticipant βάσει του διαχωριστικού «ως» που αποφασίστηκε να υλοποιηθεί. Επι της ουσίας επιστρέφει έναν Entity Individual.

getFrontFromMixString(String mixedTextField, String seperator)

Κάνει ακρίβως ότι κάνει και η getFrontFromMixString(String mixedTextField) με την διαφορά στο ότι: Δίνει την δυνατότητα στον προγραμματιστή να χρησιμοποιήσει άλλο διαχωριστικό του πεδίου EntityAsParticipant εκτός από το «ως».

getTheBacktFromMixString(String mixedTextField)

Δέχεται ένα String του οποίο το περιεχόμενο είναι η τιμή της Ιδιότητας Δεδομένων με το όνομα EntityAsParticipant του Σχήματος 3.6 και για να υλοποιηθεί η λειτουργία που περιγράφεται για το πεδίο αυτό στην 3.1.1, επιστρέφει το δεξί μέρος του EntityAsParticipant βάσει του διαχωριστικού «ως» που αποφασίστηκε να υλοποιηθεί. Επί της ουσίας επιστρέφει έναν Participant Individual.

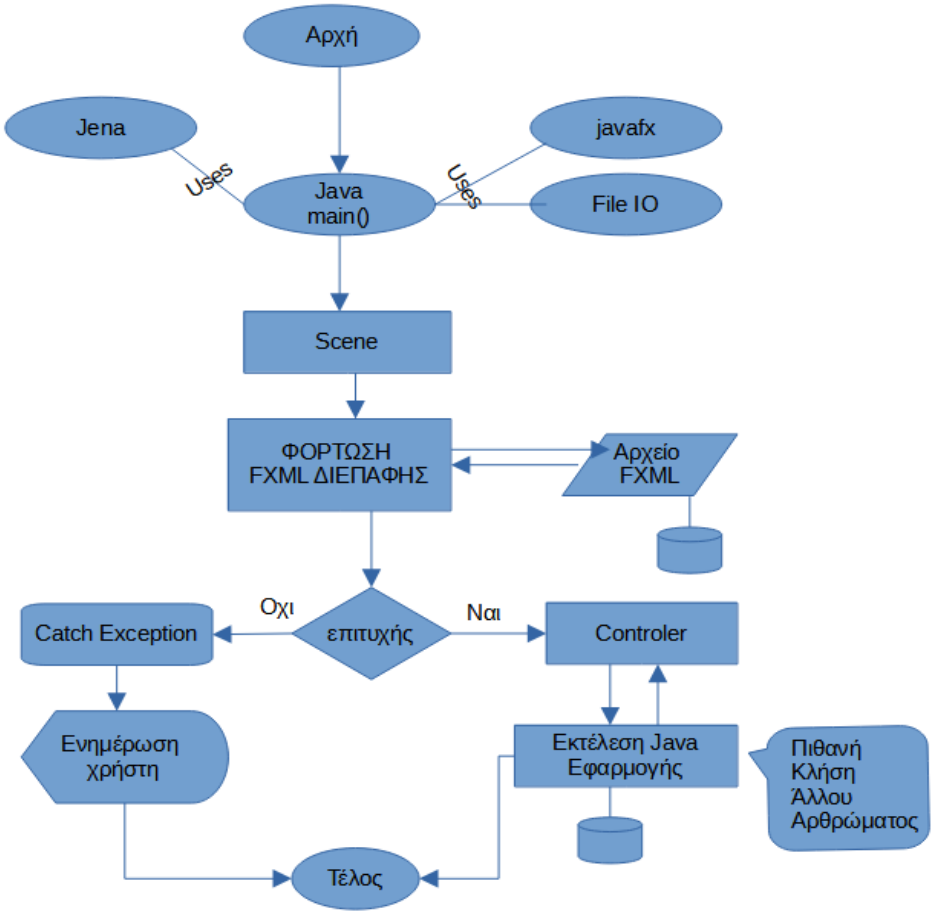
getTheBacktFromMixString(String mixedTextField, String seperator)

Κάνει ακριβώς ότι κάνει και η getTheBacktFromMixString(String mixedTextField) με την διαφορά στο ότι: Δίνει την δυνατότητα στον προγραμματιστή να χρησιμοποιήσει άλλο διαχωριστικό του πεδίου EntityAsParticipant εκτός από το «ως».

Οι παραπάνω συναρτήσεις χρησιμοποιήθηκαν και δημιουργηθήκαν κατά την ανάπτυξη της εφαρμογής και αποφασίστηκε να τοποθετηθούν σε ένα πακέτο Java για να γίνεται πιο εύκολη η χρήση τους.

3.4.5 Γενικό Διάγραμμα ροής αρθρώματος FXML

Η εφαρμογή χρησιμοποιεί την JavaFX, η βιβλιοθήκη αυτή ορίζει ένα πλαίσιο λειτουργίας που παρουσιάζεται στο *Σχήμα 3.28*. Υπάρχει ένα αρχικό Stage, μια σκηνή, που λαμβάνει χώρα η ίδια η εφαρμογή.



Σχήμα 3.28: Γενικό διάγραμμα ροής αρθρώματος

Το Stage εμφανίζει διάφορα Scene που αποτελούν το ουσιαστικό μέρος του UI (Διεπαφής Χρήστη).

Τα γραφικά στοιχεία, η δομή και η εμφάνιση του κάθε Αρθρώματος (Scene) βρίσκονται σε ένα FXML αρχείο, η διαχείριση, ενημέρωση και τροποποίηση των πληροφοριών του Αρθρώματος γίνεται από ένα αρχείο με κώδικα Java που ονομάζεται Controller.

Στον Controller εκτελούνται οι εντολές Java ή καλούνται τα κατάλληλα Java προγράμματα για την διαχείριση των δεδομένων, όταν η διαχείριση των δεδομένων επιφέρει αλλαγές στο γραφικό στοιχείο αυτές εμφανίζονται αυτές οι αλλαγές.

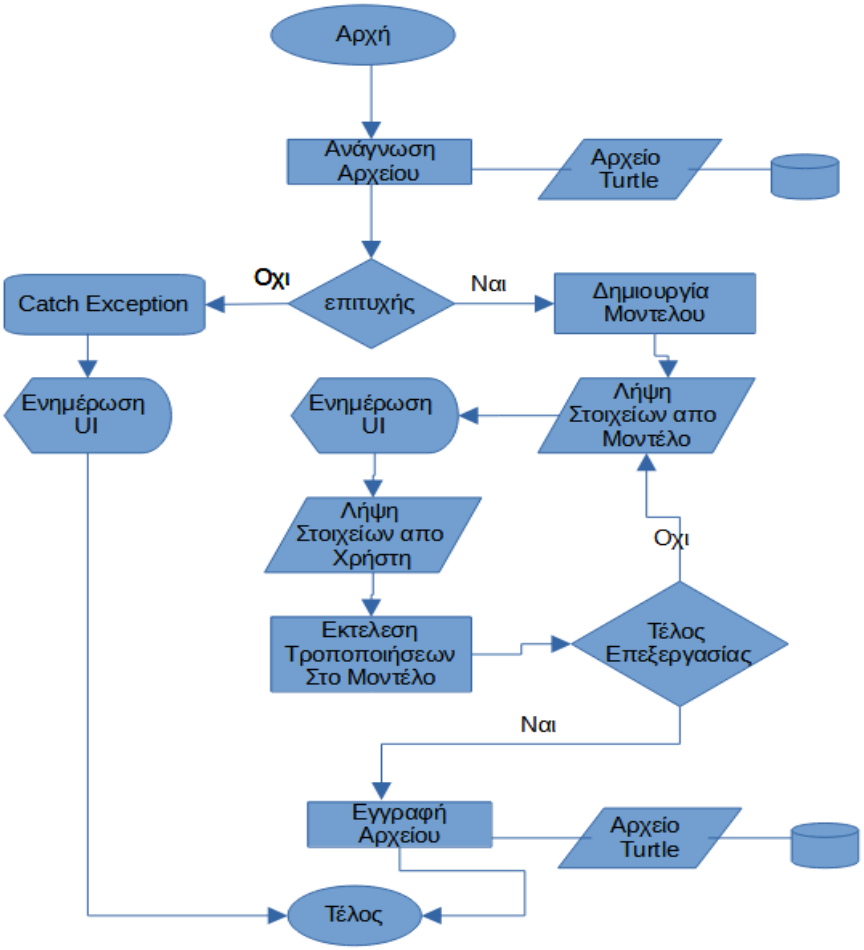
Η δομή αυτή χρησιμοποιείται σε όλο το Πληροφοριακό Σύστημα της ΔΕ, για κάθε μέρος του προγράμματος που αφορά.

Υπάρχει δηλαδή για **κάθε μέρος** της εφαρμογής ένα FXML αρχείο, ένα αρχείο Java για φόρτωση του Controller και το αρχείο Java Controller.

3.4.6 Διάγραμμα ροής επικοινωνίας με την οντολογία

Παρουσιάζεται στο *Σχήμα 3.29* το διάγραμμα ροής της επικοινωνίας της εφαρμογής με την οντολογία

Γίνεται η χρήση του Model της Jena, δηλαδή η οντολογία που βρίσκεται σε ένα αρχείο στον δίσκο με σύνταξη Turtle φορτώνεται σε ένα Μοντέλο, στο μοντέλο αυτό γίνονται ερωτήματα SPARQL, τα ερωτήματα αυτά δεν είναι μόνο για λήψη στοιχείων αλλά είναι και Update και Delete.



Σχήμα 3.29: Διάγραμμα ροής επικοινωνίας με οντολογία

Αφού τελειώσει η επικοινωνία με τον χρήστη και γίνουν οι επιθυμητές ενημερώσεις ή αλλαγές σε οθόνη και στοιχεία οντολογίας, τότε και μόνο τότε εγγράφεται το Μοντέλο στον δίσκο και ενημερώνεται το αρχείο.

3.4.7 Παράδειγμα υλοποίησης

Παρουσιάζεται η αρχική οθόνη της εφαρμογής στην οποία:

- Γίνεται χρήση FXML
- Υπάρχει Controller
- Γίνεται κλήση άλλων αρθρωμάτων της εφαρμογής
- Διαβάζεται η Οντολογία από τον δίσκο
- Λαμβάνονται στοιχεία από Μοντέλο
- Εμφανίζονται τα στοιχεία από το Μοντέλο

Στην *Εικόνα 3.30* παρουσιάζεται η αρχική οθόνη της εφαρμογής, υπάρχει ένα πάνω μενού και ένα dashboard που εμφανίζονται μερικά προεπιλεγμένα στοιχεία της οντολογίας. Γίνεται αναγνώριση των διαστάσεων της ενεργής οθόνης που εκτελείται το πρόγραμμα και οι διαστάσεις ορίζονται σύμφωνα με αυτές, έχει επιλεγεί να απενεργοποιηθούν τα κουμπιά σμίκρυνσης, επαναφοράς και κλεισίματος και η έξοδος από την εφαρμογή γίνεται αποκλειστικά από το menu Αρχείο Δεδομένων → Έξοδος.

Τα στοιχεία που εμφανίζονται είναι από το αρχείο που δημιουργήθηκε στο 3.2.1 και βρίσκεται στο Παράρτημα Β.

Εικόνα 3.30: Αρχική οθόνη ΠΣ

Αρχείο FXML

Η δομή της οθόνης δηλώνεται από το FXML αρχείο «splash2.fxml» (παρατίθεται στο παράρτημα C) και δημιουργήθηκε με το εργαλείο Scene Builder (παράγραφος 2.8.2). Παρατηρείται στον κώδικα του «splash2.fxml» στην δήλωση του VBox, *Εικόνα 3.31*, η ιδιότητα «fx:controller».

```
<VBox fx:id="VBsplash" prefHeight="531.0" prefWidth="709.0"
      xmlns="http://javafx.com/javafx/8.0.171" xmlns:fx="http://javafx.com/fxml/1"
      fx:controller="beta003.Splash2Control">
```

Εικόνα 3.31: Δήλωση VBox στο splash2.fxml

Η ιδιότητα αυτή ενημερώνει την εφαρμογή για το που βρίσκεται ο Controller της οθόνης αυτής, ο Controller είναι ένα αρχείο κώδικα Java, στην προκειμένη περίπτωση είναι το αρχείο «beta003.Splash2Control.java» και θα κληθεί (σύμφωνα με το διάγραμμα του σχήματος 3.28) να «τρέξει» τον κώδικα του.

Αρχείο Controller.java

Στο αρχείο Splash2Control.java (Παράρτημα C) εκτελούνται εντολές σύμφωνα με το διάγραμμα ροής του *Σχήματος 3.29*.

Ανάγνωση μοντέλου από το αρχείο με την χρήση της Jena **ModelFactory.createOntologyModel** [42]

Λήψη στοιχείων από το Μοντέλο για

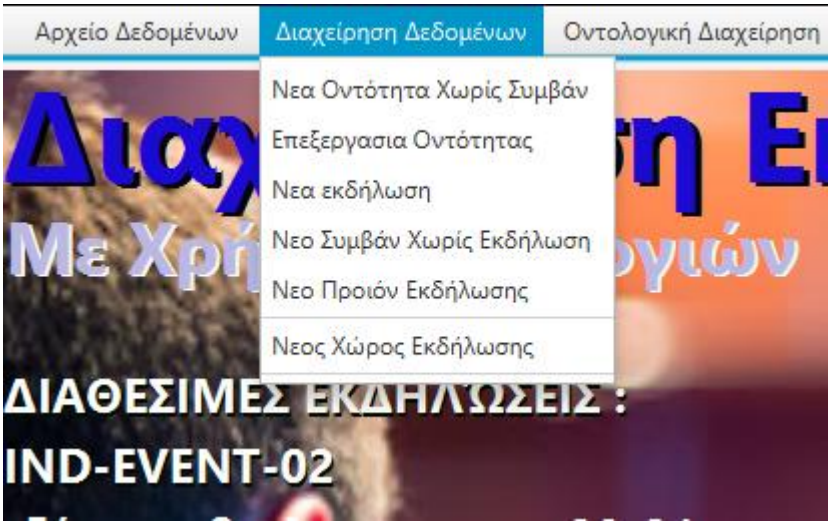
- Εκδηλώσεις με την συνάρτηση getAllIndividualsOfClass(theModel, "Event")

- Συμβάντα με την συνάρτηση getAllIndividualsOfClass(theModel, " Happening ")
- Οντότητες με την συνάρτηση getAllIndividualsOfClass(theModel, " Entity ")

Ενημέρωση διεπαφής με τα δεδομένα του Μοντέλου

Τα στοιχεία αυτά αποθηκεύονται σε Array List και κάνοντας χρήση των αντικειμένων του αρχείου «splash2.fxml» γίνεται η ενημέρωση του περιεχομένου των **FXML Label** lblM1, lblM2 και lblM3 με τα αντίστοιχα Array List. Γίνεται επαναφόρτωση της αρχικής οθόνης για να εμφανιστούν τα νέα περιεχόμενα.

Λήψη επιλογής χρήστη



Εικόνα 3.32: Menu επιλογής

Στο αρχείο «splash2.fxml» δηλώνεται το Menu της εφαρμογής το οποίο παρουσιάζεται στην Εικόνα 3.32. Παρατίθεται μέρος του κώδικα FXML της δήλωσης του Menu «**Διαχείριση Δεδομένων**»:

```
<Menu fx:id="menu_data" mnemonicParsing="false" text="Διαχείριση Δεδομένων">
<items>
        <MenuItem
        fx:id="addNewHappeningNoEvent"
        mnemonicParsing="false"
        onAction="#doAddNewEntityNoHappening"
        text="Νεα Οντότητα Χωρίς Συμβάν" />
         …
 </items>
</Menu>
```

Η δήλωση της ιδιότητας **onAction** προκαλεί την εκτέλεση της συνάρτησης που έχει δηλωθεί: **doAddNewEntityNoHappening** η συνάρτηση αυτή βρίσκεται στο **Splash2Control.java** αρχείο.

Κλήση νέου αρθρώματος εφαρμογής

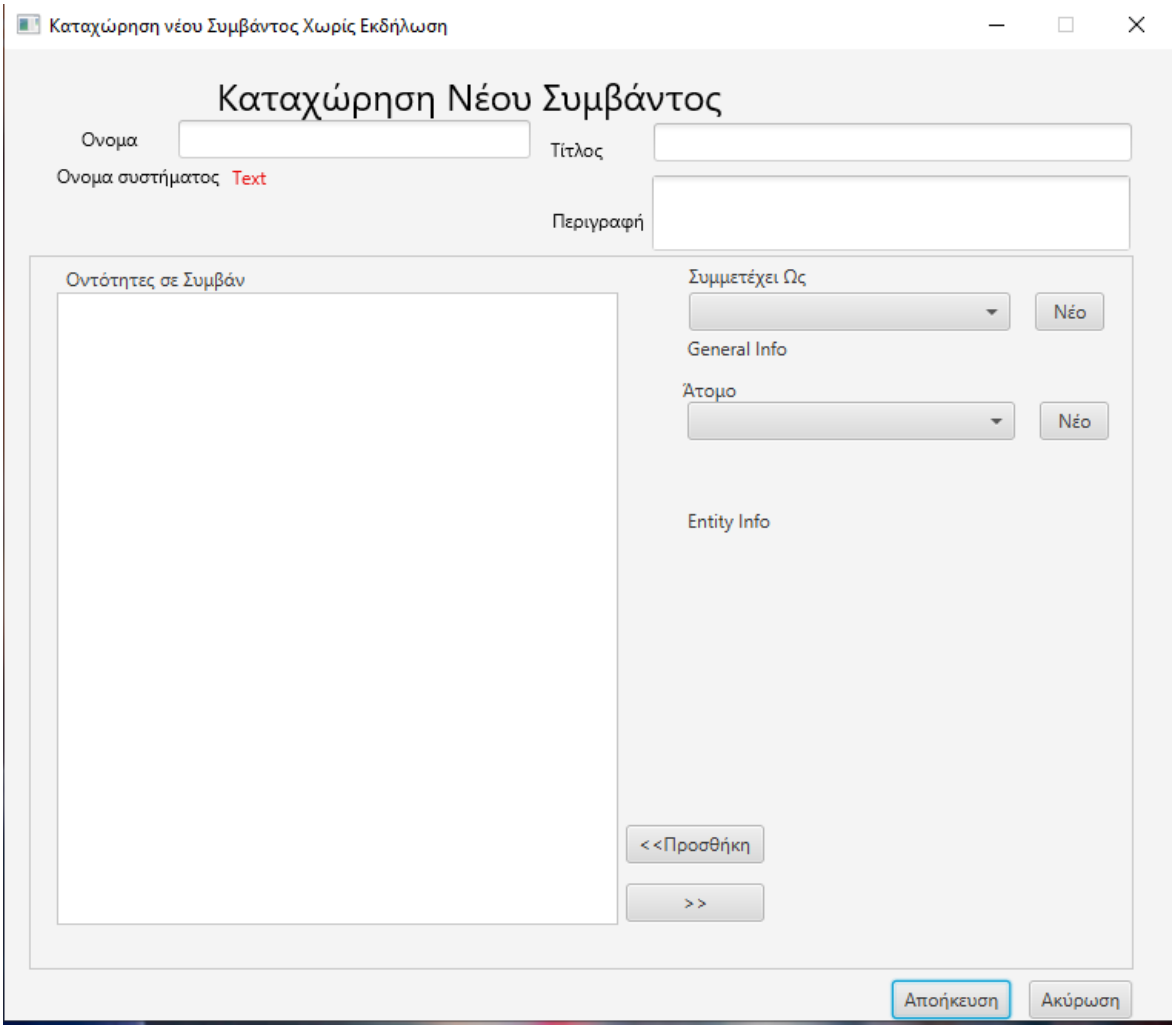
Η εκτέλεση του κώδικα της συνάρτησης doAddNewEntityNoHappening() δημιουργεί ένα νέο αντικείμενο της κλάσης **AddNewEntityNoHappening** (νέο άρθρωμα) και κατόπιν προκαλεί την ροή που παρουσιάζεται στο Σχήμα 3.28 *Γενικό διάγραμμα ροής αρθρώματος*, καλώντας την φόρτωση του αντίστοιχου FXML αρχείου «**addNewEntityNoHappening.fxml**» το οποίο με την σειρά του ορίζει τον νέο Controller «**AddNewEntityNoHappeningControl.java**».

Σε όλη την υλοποίηση του ΠΣ ακολουθούνται αυτές ακριβώς οι διαδικασίες για το διάβασμα της οντολογίας και την αλλαγή αρθρωμάτων.

3.4.8 Παράδειγμα Update

Για την ανάλυση επιλέχθηκε το άρθρωμα «**Νέο Συμβάν**» γιατί οι λειτουργίες που θα παρουσιαστούν καλύπτουν όλες τις περιπτώσεις χρήσης που συναντάμε και στην υπόλοιπη εφαρμογή, επίσης περιέχει το Data Object, «**EntityAsIndividual**» που γίνεται προγραμματιστικά σύνθετο.

Η διαδικασία του UI είναι ίδια με το παράδειγμα 3.4.5. στην εικόνα 3.33 παρουσιάζεται η οθόνη «**Καταχώρηση Νέου Συμβάντος**».



Εικόνα 3.33: Άρθρωμα Καταχώρηση νέου συμβάντος

Όνομα – Όνομα συστήματος

Επιλέχθηκε να είναι **μέρος της πληροφορίας και το URI**, καταργώντας τα πεδία **ID** του σχήματος *2.1 Αρχική Υλοποίηση UML*. Αυτό δημιουργεί την αναγκαιότητα της εξασφάλισης της μοναδικότητας του URI, για να παίξει τον **ρόλο του μοναδικού κλειδιού στο ΠΣ**. Σημασιολογικά δεν υπάρχει κανένας περιορισμός στην χρήση Ελληνικών χαρακτήρων σε ένα URI αλλά υπάρχει περιορισμός στους ειδικούς χαρακτήρες που μπορεί να εισάγει ο χρήστης, πέραν τούτου αποφασίστηκε να τεθούν κάποιοι υποκειμενικοί κανόνες που θα περιορίζουν το τι μπορεί να περάσει ο χρήστης.

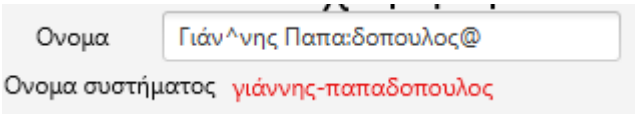
Το πεδίο FXML είναι το **txtIdivName** για το όνομα και αποφασίστηκε να υπάρχει ενημέρωση του χρήστη την στιγμή της πληκτρολόγησης για τους κανόνες αυτούς, επιδεικνύοντας το **messageSHOW που** είναι το FXML όνομα της ετικέτας ενημέρωσης του χρήστη για το όνομα που τελικά θα καταχωρηθεί στο URI. Η συνάρτηση updateMessageSHOW(KeyEvent event) αναλαμβάνει αυτήν την διαδικασία.

updateMessageSHOW

Λαμβάνεται ότι υπάρχει στο πεδίο **txtIdivName** κάθε φορά που μεταβάλλεται το περιεχόμενο του, το περνάει από την συνάρτηση χρήστη **makeLegal** η οποία, κάνει όλους τους χαρακτήρες μικρούς, εξαφανίζει τους ειδικούς χαρακτήρες και τοποθετεί μια παύλα όπου υπάρχει κενό ή tab (απλά γιατί έτσι επιλέχθηκε στην σχεδίαση), για λόγους ομοιομορφίας.

Διαβάζεται από το Μοντέλο με την συνάρτηση χρήστη **getAllUserNames(theModel)** και δημιουργείται ένα AraayList με **οτιδήποτε έχει καταχωρηθεί στην Οντολογία** σαν URI ή αποτελεί συστατικό του συντακτικού της Οντολογίας. Αν βρεθεί να υπάρχει το περιεχόμενο του **txtIdivName** στο Array List τότε προστίθεται στο τέλος (APPEND) του ονόματος URI που δίνει ο χρήστης το String «-A-COPY»

Ταυτόχρονα ενημερώνεται το Label **messageSHOW** για ενημέρωση του χρήστη σχετικά με τις αλλαγές που γίνονται.



Εικόνα 3.34: URI έλεγχος

Ο έλεγχος που πού παρουσιάζεται στην Εικόνα 3.34 εφαρμόζεται σε όλες τις οθόνες του ΠΣ με την μόνη διαφορά τις περιπτώσεις καταχώρησης κλάσης από τον χρήστη όπου δεν γίνονται οι χαρακτήρες μικροί (lower case).

Τίτλος – Περιγραφή

Τα FXML πεδία **txtHappTitle** και **txtHappTitle** αντίστοιχα λαμβάνουν την πληροφορία από τον χρήστη. Σε περίπτωση που είναι κάποιο από αυτά κενό, απενεργοποιείται το κουμπί «**Αποθήκευση**» και εμφανίζεται μήνυμα στον χρήστη. Κατά την διαδικασία καταχώρησης με το κουμπί «**Αποθήκευση**» αφαιρούνται οι νέες γραμμές και τα tab με την συνάρτηση χρήστη **noNewLinesNoTabs**.

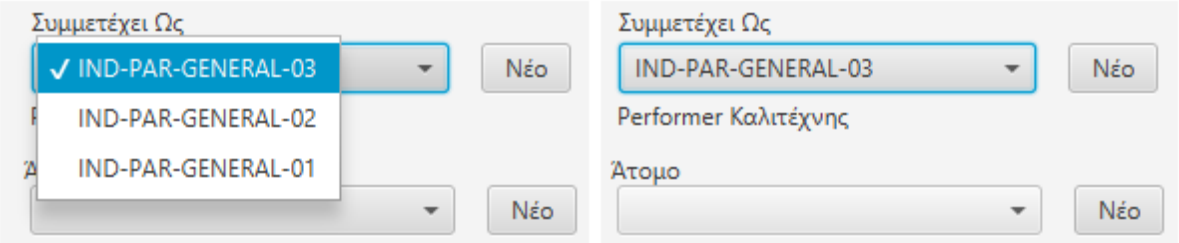
Η διαδικασία αυτών των δύο πεδίων εφαρμόζεται σε όλες τις οθόνες του ΠΣ.

Συμμετέχει ως

Το FXML Choice Box **CBGeneral** αποτελεί το σημείο που θα επιλέξει ο χρήστης το είδος (ρόλο) της συμμετοχής που θα έχει το άτομο στο συμβάν. Το περιεχόμενο του, ενημερώνεται από την οντολογία με την συνάρτηση χρήστη **getAllIndividualsOfClass(theModel,"General")**, αποφασίστηκε να εμφανίζονται **μόνο τα Individual της κλάσης General** για λόγους ομοιομορφίας. Εφόσον βρέθηκαν στοιχεία, η οποιαδήποτε δράση γίνεται στο Choice Box **CBGeneral** προκαλεί την κλήση μιας ανώνυμης συνάρτησης.

Ανώνυμη συνάρτηση

Λαμβάνει την επιλογή χρήστη και ψάχνει στην Οντολογία για να φέρει τις πληροφορίες για τον Individual που επιλέχθηκε με την συνάρτηση χρήστη **getValuesOfPropertiresOfIndividual**, ενημερώνει την FXML ετικέτα **lblGeneralInfo** για να εμφανιστούν οι πληροφορίες κάτω από το Choice Box **CBGeneral** η λειτουργία αυτή παρουσιάζεται στην *Εικόνα 3.35*.



Εικόνα 3.35: UI Choice Box «Συμμετέχει Ως»

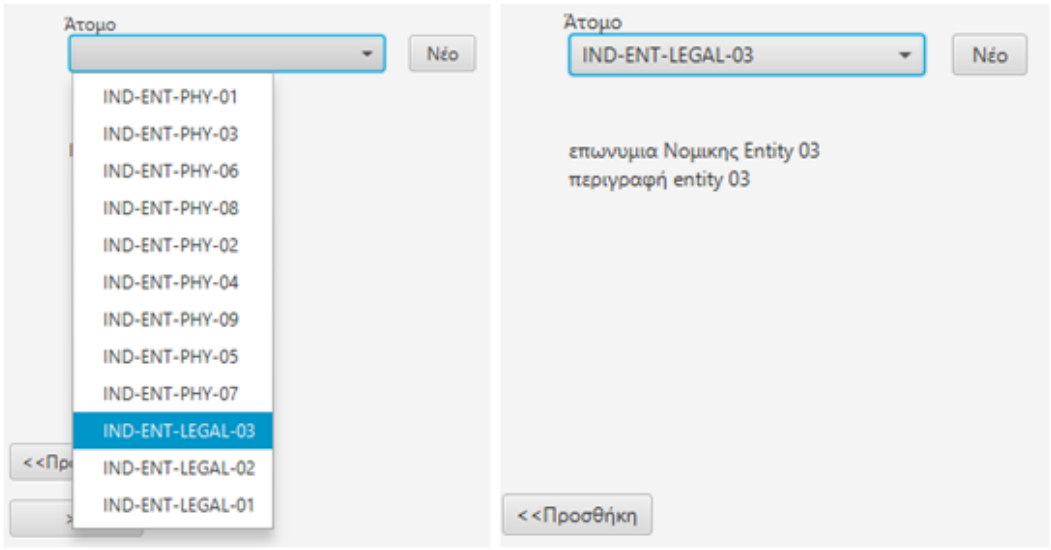
Η διαδικασία του Choice Box εφαρμόζεται με τον ίδιο τρόπο σε όλες τις οθόνες του ΠΣ

Άτομο

Το FXML Choice Box **CBEntities** αποτελεί το σημείο που θα επιλέξει ο χρήστης το Άτομο (Entity) που θα συμμετέχει στο Happening. Το περιεχόμενο του, ενημερώνεται από την Οντολογία με την συνάρτηση χρήστη **getAllIndividualsOfSuperClass(theModel,"Entity")**, αποφασίστηκε να εμφανίζονται **όλα Individual της υπερκλάσης Entity** διότι ο συμμετέχοντας μπορεί να είναι και μη φυσικό πρόσωπο. Εφόσον βρέθηκαν στοιχεία, η οποιαδήποτε δράση γίνεται στο Choice Box **CBEntities** προκαλεί την κλήση μιας ανώνυμης συνάρτησης.

Ανώνυμη συνάρτηση

Με τον ίδιο τρόπο λαμβάνει την επιλογή χρήστη και ψάχνει στην οντολογία για να φέρει τις πληροφορίες για τον Individual που επιλέχθηκε με την συνάρτηση χρήστη **getValuesOfPropertiresOfIndividual**, ενημερώνει την FXML ετικέτα **lblEntityInfo** για να εμφανιστούν οι πληροφορίες κάτω από το Choice Box Box **CBEntities** η λειτουργία αυτή παρουσιάζεται στην *Εικόνα 3.36*.



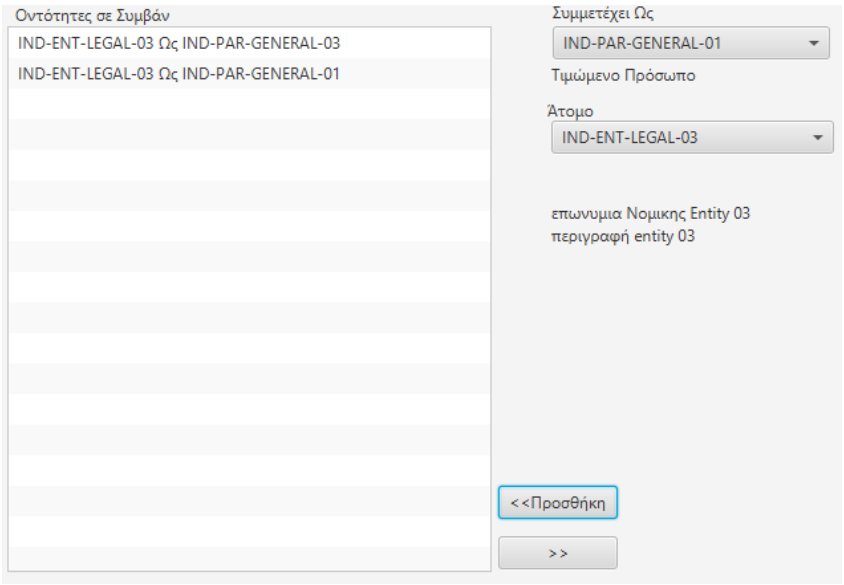
Εικόνα 3.36: UI Choice Box άτομο

Άτομο σε συμβάν

Το FXML List View αντικείμενο **LVEntitiesInHapp** είναι η υλοποίηση προγραμματιστικά του «Σύνθετου» πεδίου **EntityAsParticipant** (*Σχήμα 3.6 Διάγραμμα κλάσης Happening*), με το πάτημα του κουμπιού «**Προσθήκη**» εκτελείται η συνάρτηση **addHappeningToList**, το κουμπί είναι ενεργό **μόνο εάν** ο χρήστης έχει κάνει επιλογή «Συμμετέχει Ως» και «Άτομο», Παρουσιάζεται την *Εικόνα 3.36.*

addHappeningToList

Γίνεται έλεγχος αν υπάρχει ήδη ο συνδυασμός του ατόμου με τον τρόπο συμμετοχής, αν δεν υπάρχει τότε λαμβάνεται η επιλογή του Ατόμου (URI) που έγινε στο **CBEntities** από τον χρήστη και δημιουργείται ένα String με την πληροφορία αυτή ακολουθούμενη από ένα κενό, την λέξη «Ως» και ένα ακόμα κενό. Κατόπιν λαμβάνεται η πληροφορία(URI) του είδους συμμετοχής από το **CBGeneral** και τοποθετείται στο τέλος του ιδίου String (μετά το «Ως »).



Εικόνα 3.37: UI List View άτομο σε συμβάν

String row = CBEntities.getValue().toString()+" Ως "+CBGeneral.getValue().toString();

Το String που δημιουργήθηκε γίνεται προσθήκη στο τέλος της List View **LVEntitiesInHapp**

Η ίδια διαδικασία επαναλαμβάνεται για όσο υπάρχει επιλεγμένο άτομο και ρόλος συμμετοχής και πατά ο χρήστης το κουμπί «**Προσθήκη**».

Με την χρήση του «Σύνθετου» πεδίου δίδεται η δυνατότητα, όπως φαίνεται στην Εικόνα 3.37, ***το ίδιο άτομο να είναι και «Performer Καλλιτέχνης» και «Τιμώμενο πρόσωπο»*** *χωρίς να εξαναγκάζεται ένας Reasoner να συμπεράνει ότι οι ιδιότητες «Performer Καλλιτέχνης» και «Τιμώμενο πρόσωπο» είναι ίδιες.*

Έχοντας επιλέξει ο χρήστης μία από τις επιλογές του List View «Οντότητες σε Συμβάν» γίνεται η ενεργοποίηση του **κουμπιού « >> »**, το οποίο αφαιρεί το επιλεγμένο πεδίο.

Αποθήκευση

Εφόσον έχουν ικανοποιηθεί όλες οι συνθήκες έλεγχου που περιγράφονται παραπάνω τότε ενεργοποιείται το κουμπί «Αποθήκευση», το πάτημα του κουμπιού δημιουργεί το κατάλληλο Update Query και καλεί την συνάρτηση χρήστη **updateQueryOnTTLFile** η οποία ενημερώνει το αρχείο της οντολογίας με τις νέες τριάδες (εγγραφές) που δημιούργησε το άρθρωμα «**Νέο Συμβάν**». Η νέα πληροφορία είναι πλέον διαθέσιμη σε όλο το ΠΣ.

Σε όλη την υλοποίηση του ΠΣ ακολουθούνται αυτές ακριβώς οι διαδικασίες για την ενημέρωση της Οντολογίας, το διάβασμα και «γέμισμα» List, για την εμφάνιση στοιχειών Individual και για τον περιορισμό και έλεγχο της συμπεριφοράς του χρήστη.

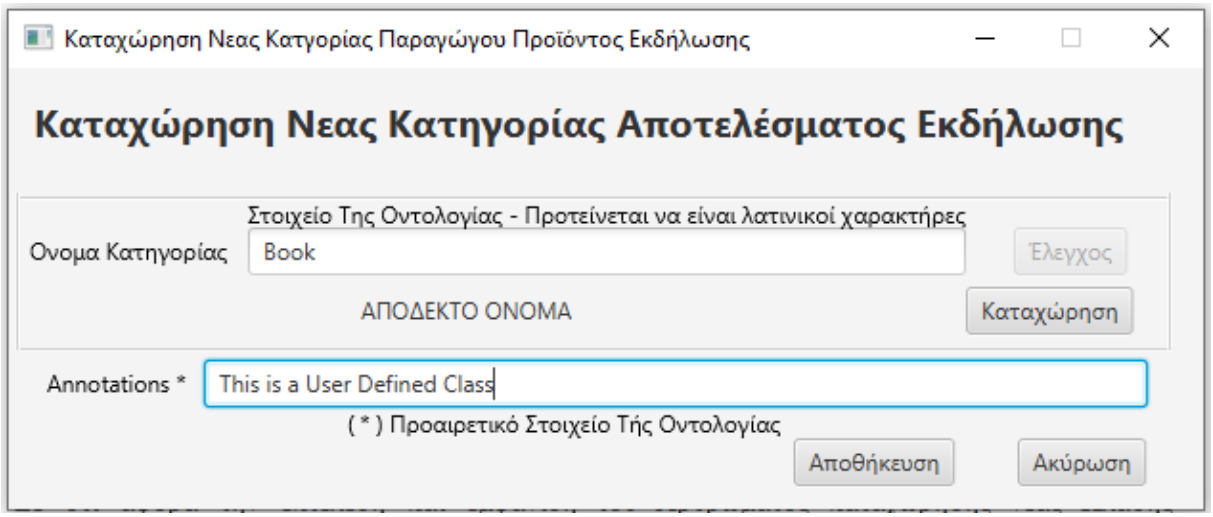
3.4.9 Παράδειγμα καταχώρησης νέας κλάσης

Αποφασίστηκε να δίνεται η δυνατότητα στον Χρήστη του ΠΣ να δημιουργεί κλάσεις, υποκλάσεις στην ουσία, κάποιων κατηγοριών. Οι κλάσεις (αυτές που επιτρέπεται να έχουν υποκλάσεις ορισμένες από τον χρήστη) είναι οι **Participant**, **Product** και **Scope**, τα αντίστοιχα διαγράμματα τους είναι τα *Σχήματα 3.3, 3.5 και 3.7.*

Επιδιώχθηκε να έχουν την ίδια δομή, με μόνη εξαίρεση την κλάση Participant που δεν έχει πεδίο περιγραφής.

Σε ότι αφορά την εκτέλεση και εμφάνιση του αρθρώματος καταχώρησης νέας κλάσης ακολουθούνται οι ίδιες διαδικασίες του *Σχήματος 3.27 Διάγραμμα ροής αρθρώματος* και του παραδείγματος 3.4.5.

Η επικοινωνία με την οντολογία γίνεται με την ίδια ροή του *Σχήματος 3.29 διάγραμμα ροής επικοινωνίας με οντολογία,* και με παρόμοιες διαδικασίες με το παράδειγμα 3.4.6 για το «γέμισμα» των **FXML Choice Box**, τον έλεγχο των κουμπιών και την διαδικασία αποθήκευσης (ενημέρωσης) του αρχείου της οντολογίας.



Εικόνα 3.38: UI Καταχώρηση νέας κατηγορίας παραγώγου εκδήλωσης

Διαφορές

Η πρώτη διαφορά σε σχέση με τις προηγούμενες διαδικασίες είναι το ότι χρησιμοποιείται η Οντολογική Ιδιότητα **rdfs:comment** για αποθήκευση πληροφορίας σχετικά με την νέα κλάση, ώστε να υπάρχει πληροφόρηση που εμφανίζεται σε περιβάλλοντα όπως το Protégé. Στην *Εικόνα 3.38* παρουσιάζεται η διεπαφή χρήστη.

Η άλλη διαφορά είναι στον έλεγχο της εισόδου του χρήστη στο πεδίο «**Όνομα Κατηγορίας**» που είναι το URI της νέας υποκλάσης. Γίνεται έλεγχος για ειδικούς χαρακτήρες αλλά δεν αφαιρούνται αυτόματα και γίνεται έλεγχος για την ύπαρξη άλλης κλάσης με το ίδιο όνομα αλλά δεν τροποποιείται το όνομα καθόλου. Γίνεται ενημέρωση του χρήστη μέσω του κουμπιού «**Έλεγχος**» για το αν είναι αποδεκτό το νέο (URI) όνομα κλάσης με τα FXML Label **lblSysInfo** και **lblOK**. Εφόσον δώσει ο χρήστης αποδεκτό URI ενεργοποιούνται το πεδίο annotation και το κουμπί «Αποθήκευση» και απενεργοποιείται το πεδίο «Όνομα Κατηγορίας» και το κουμπί «Έλεγχος».

3.5 Νέα δεδομένα ελέγχου ΠΣ

Αποφασίστηκε να γίνει εισαγωγή νέων δεδομένων για έλεγχο του ΠΣ χωρίς να διαγραφούν τα δεδομένα ελέγχου των παραγράφων 3.1.2 και 3.1.3 που έγιναν εισαγωγή για την δοκιμή της Οντολογίας στο Protégé και στο Virtuoso.

Τα νέα δεδομένα ελέγχου θα καταχωρηθούν αποκλειστικά και μόνο από την διεπαφή του ΠΣ και θα εξαχθούν δύο αρχεία, το ένα με σύνταξη OWL και το άλλο με σύνταξη RDF/XML βάσει της επιλογής που υλοποιήθηκε να δίνεται στον χρήστη.

Σε αυτά τα δύο αρχεία θα εξεταστεί αρχικά η μορφή τους με το «μάτι» του ανθρώπου για να διαπιστωθεί αν η δομή και η εμφάνιση συμφωνεί με τα προσδοκώμενα αποτελέσματα.

Κατόπιν θα δοκιμαστεί το άνοιγμα τους από το Protégé και θα παρατηρηθεί η εμφάνιση και η συμπεριφορά της Οντολογίας σε σχέση με τα αρχεία που δημιουργήθηκαν από το ΠΣ μας.

Θα ανεβεί το RDF/XML στο OpenLink Virtuoso Server, θα τεθούν ερωτήματα SPARQL που θα εμπλέκουν τα δεδομένα ελέγχου.

Θεωρήθηκε ότι πρέπει να υλοποιηθούν τα παρακάτω συστατικά:

- Καταχώρηση νέας κλάσης και στις τρείς κλάσεις που δίδεται η δυνατότητα.
- Καταχώρηση Individual για κάθε μια από αυτές τις νέες κλάσεις.
- Καταχωρηθούν Individual όλες τις άλλες κλάσεις.

Υλοποιήσεις:

- Κλάση ThesisPresentation υποκλάση της Scope, δεν θα έχει υλοποίηση, θα ορίζει μια εκδήλωση
- Κλάση AcademicPublication υποκλάση της Product
- Individual “δημοσίευση-διπλωματικής” υλοποίηση της κλάσης AcademicPublication
- Κλάση Professor υποκλάση της Pparticipant
- Individual “εισηγητής” υλοποίηση της κλάσης Professor
- Individual “εξεταστής” υλοποίηση της κλάσης Professor
- Κλάση Student υποκλάση της Pparticipant
- Individual “προπτυχιακός” υλοποίηση της κλάσης Student
- Individual “μεταπτυχιακός” υλοποίηση της κλάσης Student
- Individual “allios-yiannis” υλοποίηση της κλάσης Physical
- Individual “karipidis-mike” υλοποίηση της κλάσης Physical
- Individual “moskov-pavlos” υλοποίηση της κλάσης Physical
- Individual “weis-ermolaos” υλοποίηση της κλάσης Physical
- Individual “elles-yiannis” υλοποίηση της κλάσης Physical
- Individual “moze-avraam” υλοποίηση της κλάσης Physical
- Individual “markonis-adonis” υλοποίηση της κλάσης Physical
- Individual “peninsula-of-haemus-univercity” υλοποίηση της κλάσης Legal
- Individual “conference-hall-of-pohu” υλοποίηση της κλάσης Place
- Individual “Έναρξη Παρουσίασης Εργασιών ΠΧΤΑ 20-21” υλοποίηση της κλάσης Happening
- Individual “Παρουσίαση 1 ΠΧΤΑ 20-21” υλοποίηση της κλάσης Happening
- Individual “Βαθμολόγηση 1 ΠΧΤΑ 20-21” υλοποίηση της κλάσης Happening
- Individual “Παρουσίαση 2 ΠΧΤΑ 20-21” υλοποίηση της κλάσης Happening
- Individual “Βαθμολόγηση 2 ΠΧΤΑ 20-21” υλοποίηση της κλάσης Happening
- Individual “Λήξη Παρουσίασης Εργασιών ΠΧΤΑ 20-21” υλοποίηση της κλάσης Happening
- Individual “Εξέταση-διπλωματικων-20-21” υλοποίηση της κλάσης Event

Σχέσεις:

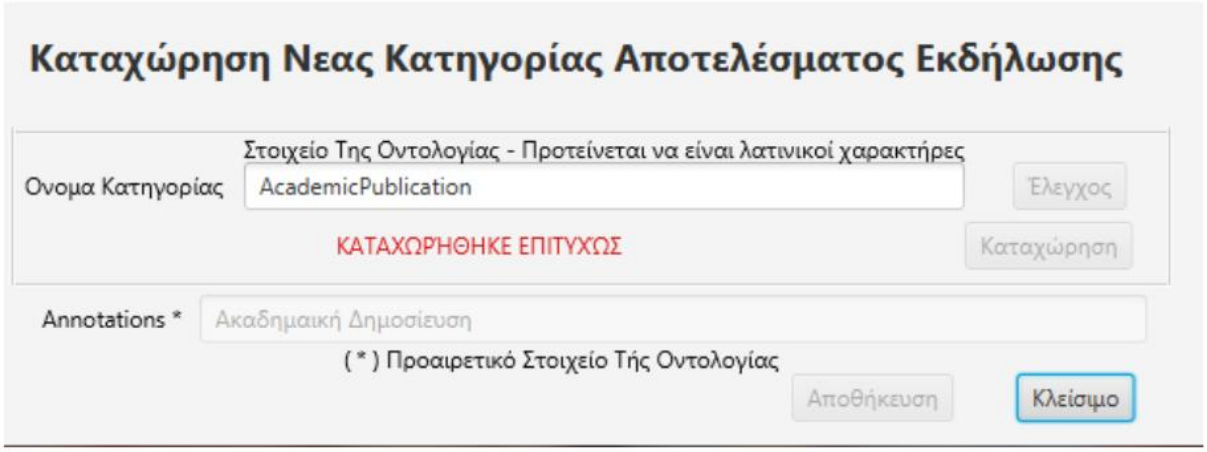
- Η “εξέταση-διπλωματικων-20-21” θα έχει τα συμβάντα
    - “Έναρξη Παρουσίασης Εργασιών ΠΧΤΑ 20-21”
    - “Παρουσίαση 1 ΠΧΤΑ 20-21”
    - “Βαθμολόγηση 1 ΠΧΤΑ 20-21”
    - “Παρουσίαση 2 ΠΧΤΑ 20-21”
    - “Βαθμολόγηση 2 ΠΧΤΑ 20-21”
    - “Λήξη Παρουσίασης Εργασιών ΠΧΤΑ 20-21”
- Η “εξέταση-διπλωματικων-20-21” θα έχει Scope ThesisPresentation
- Η “εξέταση-διπλωματικων-20-21” θα έχει Scope ThesisPresentation
- Ο “moskov-pavlos” θα συμμετέχει στο συμβάν “ Έναρξη Παρουσίασης Εργασιών ΠΧΤΑ 20-21 ως “ ομιλητής ”
- Ο “allios-yiannis” θα συμμετέχει στο συμβάν “Παρουσίαση 1 ΠΧΤΑ 20-21” ως “προπτυχιακός”
- Ο “weis-ermolaos ” θα συμμετέχει στο συμβάν “Παρουσίαση 1 ΠΧΤΑ 20-21” ως “εισηγητής”

- Ο “elles-yiannis ” θα συμμετέχει στο συμβάν “Βαθμολόγηση 1 ΠΧΤΑ 20-21” ως “εξεταστής”
- Ο “moze-avraam ” θα συμμετέχει στο συμβάν “Βαθμολόγηση 1 ΠΧΤΑ 20-21” ως “εξεταστής”
- Ο “karipidis-mike ” θα συμμετέχει στο συμβάν “Παρουσίαση 2 ΠΧΤΑ 20-21ως “μεταπτυχιακός”
- Ο “moze-avraam” θα συμμετέχει στο συμβάν “Παρουσίαση 2 ΠΧΤΑ 20-21” ως “εισηγητής”
- Ο “elles-yiannis ” θα συμμετέχει στο συμβάν “Βαθμολόγηση 2 ΠΧΤΑ 20-21 ως “εξεταστής”
- Ο “weis-ermolaos ” θα συμμετέχει στο συμβάν “Βαθμολόγηση 2 ΠΧΤΑ 20-21” ως “εξεταστής”
- Ο “ moskov-pavlos ” θα συμμετέχει στο συμβάν “Λήξη Παρουσίασης Εργασιών ΠΧΤΑ 20-21” ως “ομιλητής”
- Ο “markonis-adonis” θα συμμετέχει σε όλα τα συμβάντα ως “τεχνικός εικόνας ήχου”

3.6 Εισαγωγή δεδομένων

Καταχώρηση κλάσεων

Παρουσιάζεται στην *Εικόνα 3.39* ενδεικτικά η καταχώρηση της Κλάσης **AcademicPublication** και οι δηλώσεις Turtle που εγγράφηκαν στο αρχείο του ΠΣ.



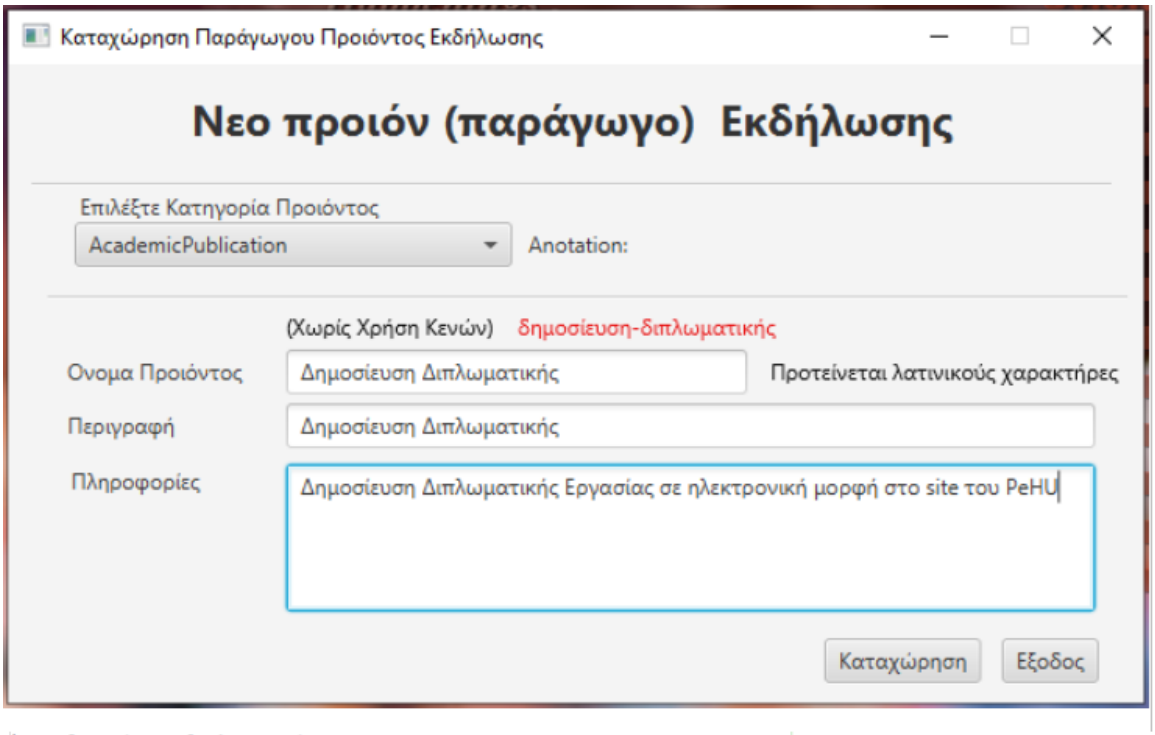

```
base:AcademicPublication
        a                owl:Class ;
        rdfs:comment     "Ακαδημαική Δημοσίευση" ;
        rdfs:subClassOf  base:Product .
```

Εικόνα 3.39: UI Καταχώρηση νέας κατηγορίας και κώδικας Turtle

Καταχωρήθηκαν και οι υπόλοιπες κλάσεις και στιγμιότυπα οθόνης μαζί με δηλώσεις Turtle παρουσιάζονται αναλυτικά στο Παράρτημα D.

Υλοποίηση Individual των νέων κλάσεων

Παρουσιάζεται στην *Εικόνα 3.40* ενδεικτικά η καταχώρηση του Individual “**δημοσίευση-διπλωματικής**” και οι δηλώσεις Turtle που εγγράφηκαν στο αρχείο του ΠΣ.



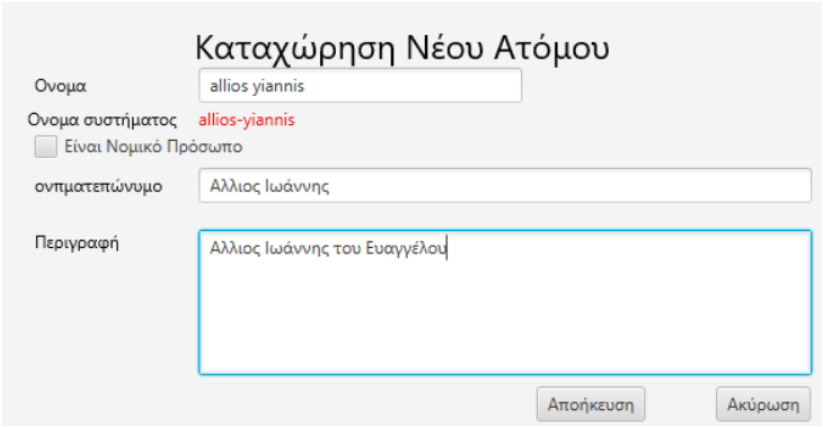

```
base:δημοσίευση-διπλωματικής
        a                       owl:NamedIndividual , base:AcademicPublication ;
        base:scopeDescription   "Δημοσίευση Διπλωματικής" ;
        base:scopeInfo          "Δημοσίευση Διπλωματικής Εργασίας σε ηλεκτρονική μορφή στο site του PeHU" .
```

Εικόνα 3.40: UI Καταχώρηση νέου Individual και κώδικας Turtle

Καταχωρήθηκαν και οι υπόλοιποι Individual των νέων κλάσεων, στιγμιότυπα οθόνης μαζί με δηλώσεις Turtle παρουσιάζονται αναλυτικά στο παράρτημα D.

Υλοποίηση Individual της κλάσης Entity

Παρουσιάζεται στην *Εικόνα 3.41* ενδεικτικά η καταχώρηση του Individual "**allios-yiannis**" και οι δηλώσεις Turtle που εγγράφηκαν στο αρχείο του ΠΣ.



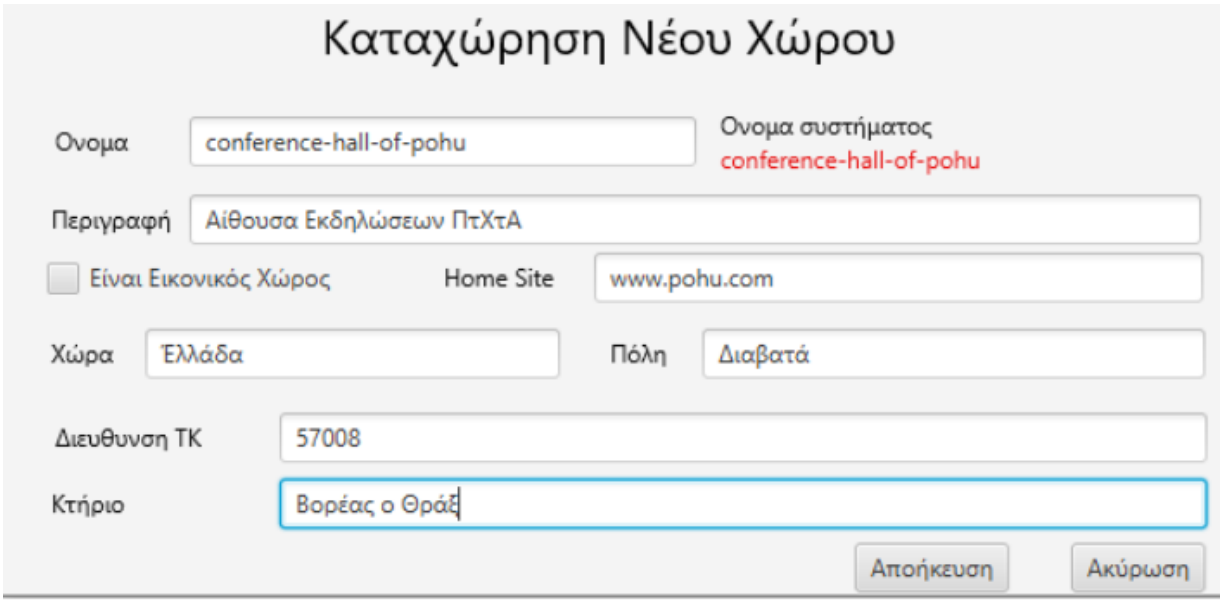

```
base:allios-yiannis  a                   owl:NamedIndividual , base:Physical ;
        base:EntityDescription   "Αλλιος Ιωάννης του Ευαγγέλου" ;
        base:EntityPhisicalName  "Αλλιος Ιωάννης" .
```

Εικόνα 3.41: UI Καταχώρηση νέου Individual Entity και κώδικας Turtle

Καταχωρήθηκαν και οι υπόλοιποι Individual των υποκλάσεων της Entity, στιγμιότυπα οθόνης μαζί με δηλώσεις Turtle παρουσιάζονται αναλυτικά στο παράρτημα D.

Υλοποίηση Individual της κλάσης Place

Παρουσιάζεται στην *Εικόνα 3.42* ενδεικτικά η καταχώρηση του Individual "**conference-hall-of-pohu**" και οι δηλώσεις Turtle που εγγράφηκαν στο αρχείο του ΠΣ.

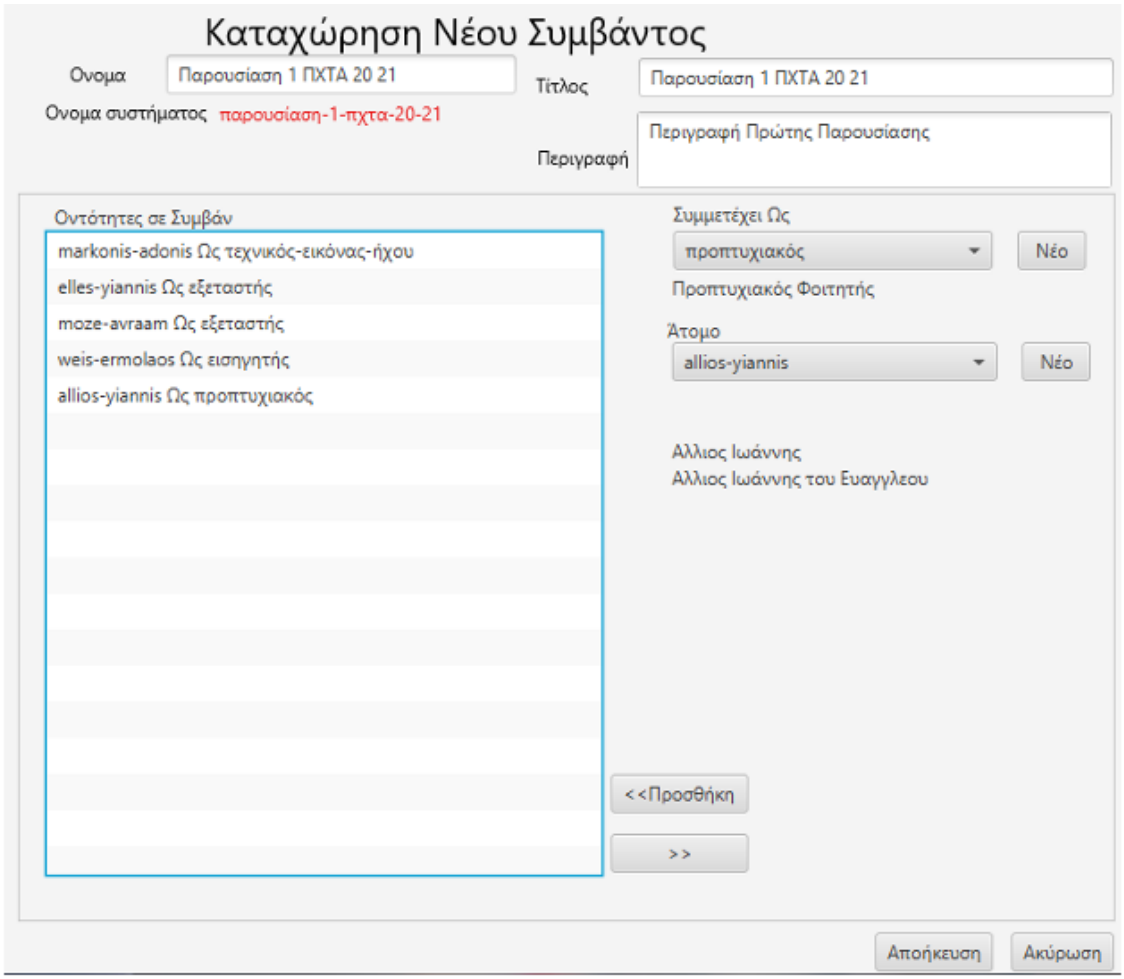



```
base:conference-hall-of-pohu
        a                   owl:NamedIndividual , base:Actual ;
        base:PlaceAddress   "57008" ;
        base:PlaceBuilding  "Βορέας ο Θράξ" ;
        base:PlaceCity      "Διαβατά" ;
        base:PlaceCountry   "Ελλάδα" ;
        base:PlaceDiscription  "Αίθουσα Εκδηλώσεων ΠτΧτΑ" ;
        base:PlaceWebAddress   "www.pohu.com" .
```

Εικόνα 3.42: UI Καταχώρηση νέου Individual Entity και κώδικας Turtle

Υλοποίηση Individual της κλάσης Happening

Παρουσιάζεται στην *Εικόνα 3.43* ενδεικτικά η καταχώρηση του Individual παρουσίαση 1 ΠΧΤΕ 20 21 ”και οι δηλώσεις Turtle που εγγράφηκαν στο αρχείο του ΠΣ.



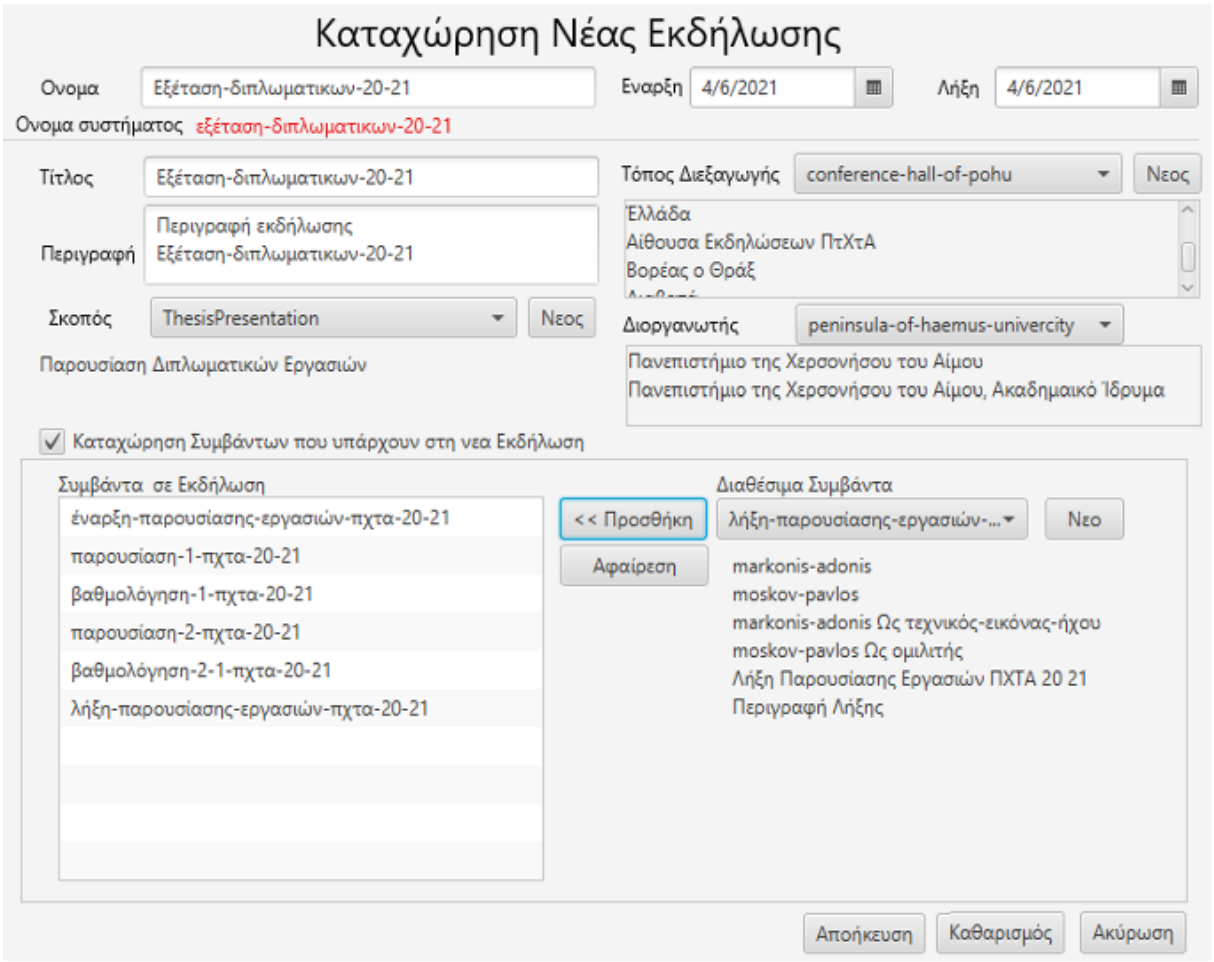

```
base:παρουσίαση-1-πχτα-20-21
        a                               owl:NamedIndividual ,
                                        base:Happening ;
        base:EntityAsParticipant        "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ,
                                        "weis-ermolaos Ως εισηγητής" ,
                                        "elles-yiannis Ως εξεταστής" ,
                                        "allios-yiannis Ως προπτυχιακός" ,
                                        "moze-avraam Ως εξεταστής" ;
        base:HappDescription            "Περιγραφή Πρώτης Παρουσίασης " ;
        base:HappInfo                   "Παρουσίαση 1 ΠΧΤΑ 20 21" ;
        base:hasAsFactor                base:moze-avraam ,
                                        base:allios-yiannis ,
                                        base:elles-yiannis ,
                                        base:weis-ermolaos ,
                                        base:markonis-adonis .
```

Εικόνα 3.43: UI Καταχώρηση νέου Individual Happening και κώδικας Turtle

Καταχωρήθηκαν και οι υπόλοιποι Individual της Happening, στιγμιότυπα οθόνης μαζί με δηλώσεις Turtle παρουσιάζονται αναλυτικά στο παράρτημα D.

Υλοποίηση Individual της κλάσης Event

Παρουσιάζεται στην *Εικόνα 3.44* η καταχώρηση του Individual "**εξέταση-διπλωματικων-20-21**" και οι δηλώσεις Turtle που εγγράφηκαν στο αρχείο του ΠΣ.

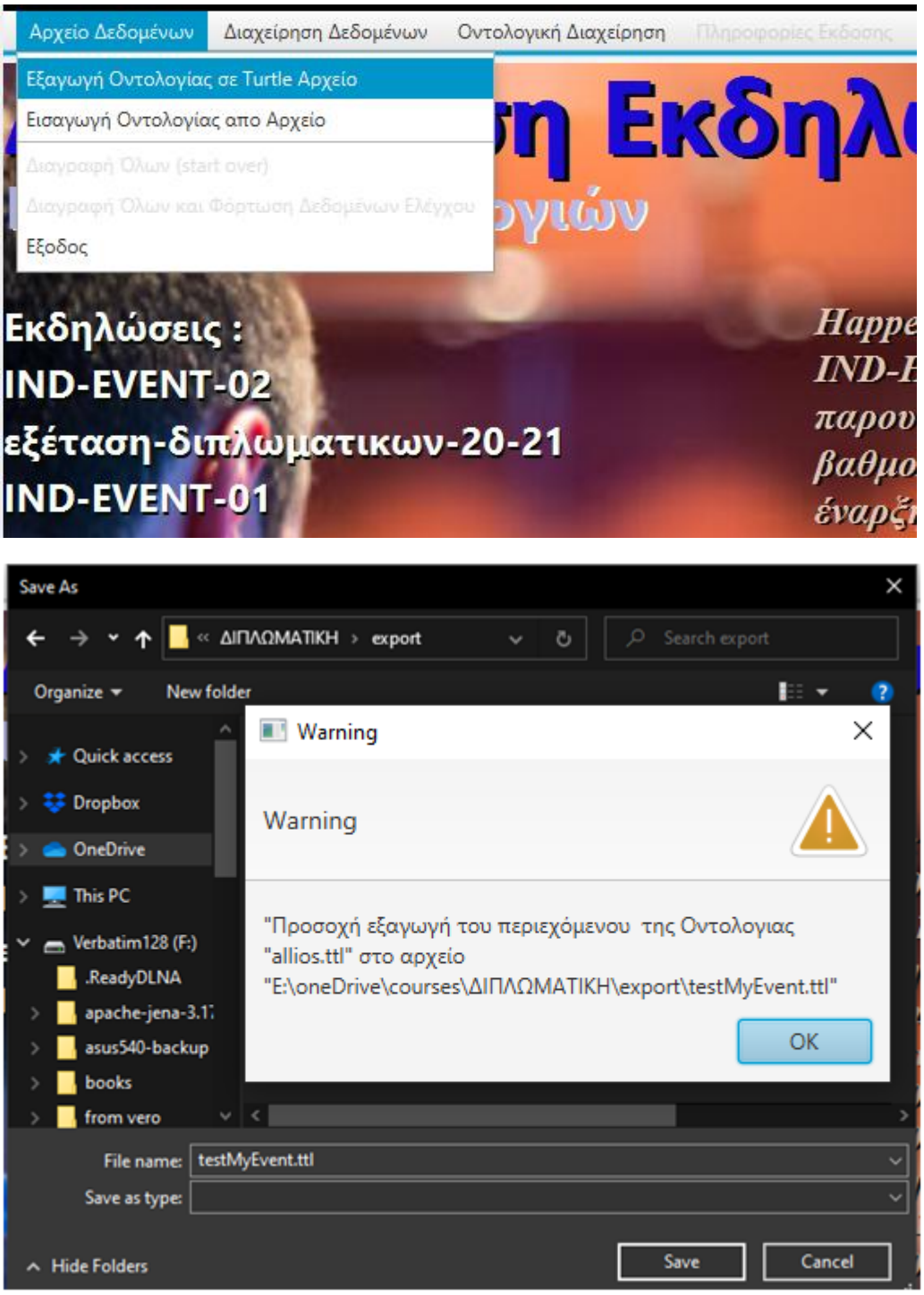



```
base:εξέταση-διπλωματικων-20-21
        a                       owl:NamedIndividual ,
                                base:ThesisPresentation ,
                                base:Event ;
        base:EventBeginDate     "2021-06-04" ;
        base:EventDescription   "Περιγραφή εκδήλωσης  Εξέταση-διπλωματικων-20-21" ;
        base:EventEndDate       "2021-06-04" ;
        base:EventTitle         "Εξέταση-διπλωματικων-20-21" ;
        base:hasHappening       base:έναρξη-παρουσίασης-εργασιών-πχτα-20-21 ,
                                base:παρουσίαση-1-πχτα-20-21 ,
                                base:παρουσίαση-2-πχτα-20-21 ,
                                base:λήξη-παρουσίασης-εργασιών-πχτα-20-21 ,
                                base:βαθμολόγηση-1-πχτα-20-21 ,
                                base:βαθμολόγηση-2-1-πχτα-20-21 ;
        base:hasHost            base:peninsula-of-haemus-univercity ;
        base:hasPlace           base:conference-hall-of-pohu .
```

Εικόνα 3.44: UI Καταχώρηση νέου Individual Event και κώδικας Turtle

3.7 Εξαγωγή δεδομένων παρατήρηση σύνταξης

Γίνεται εξαγωγή της ενημερωμένης Οντολογίας από το αντίστοιχο menu (*Εικόνα 3.45*).



Εικόνα 3.45: UI Εξαγωγή οντολογίας

Παρατηρήθηκε οπτικά και στα δύο αρχεία ότι οι εγγραφές που γίνανε από το ΠΣ φαίνονται τελείως φυσιολογικές με το ανθρώπινο μάτι, δεν επηρεάστηκαν οι εγγραφές που έγιναν με το Protégé, ερμηνεύτηκε η σύγκριση των στοιχείων του κώδικα των αρχείων, με τα δεδομένα ως αναμενόμενες και σωστές.

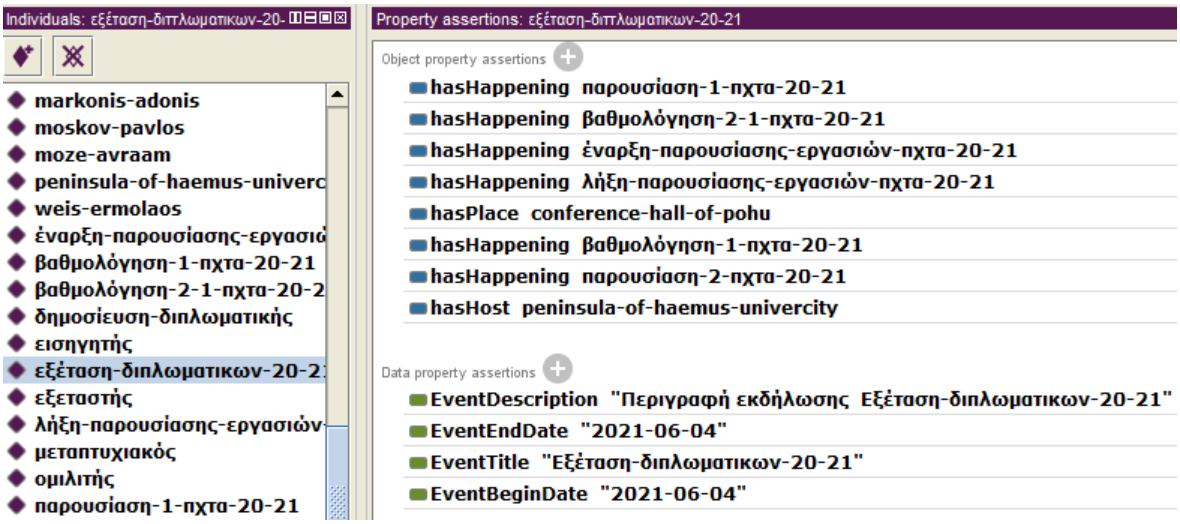
Κεφάλαιο 4ο: Συμπεράσματα

4.1 Γενικά

Με την χρήση των κατάλληλων εργαλείων μπορεί να αναπαρασταθεί η πληροφορία της οργάνωσης μιας εκδήλωσης με Σημασιολογικό τρόπο. Το ΠΣ που παρουσιάστηκε διαχειρίζεται επαρκώς τις πληροφορίες που επιλέχθηκαν να συμπεριληφθούν, παράγει Οντολογία που είναι αποδεκτή σε εργαλεία ελέγχου και είναι ανοικτό σε επέκταση και γενίκευση.

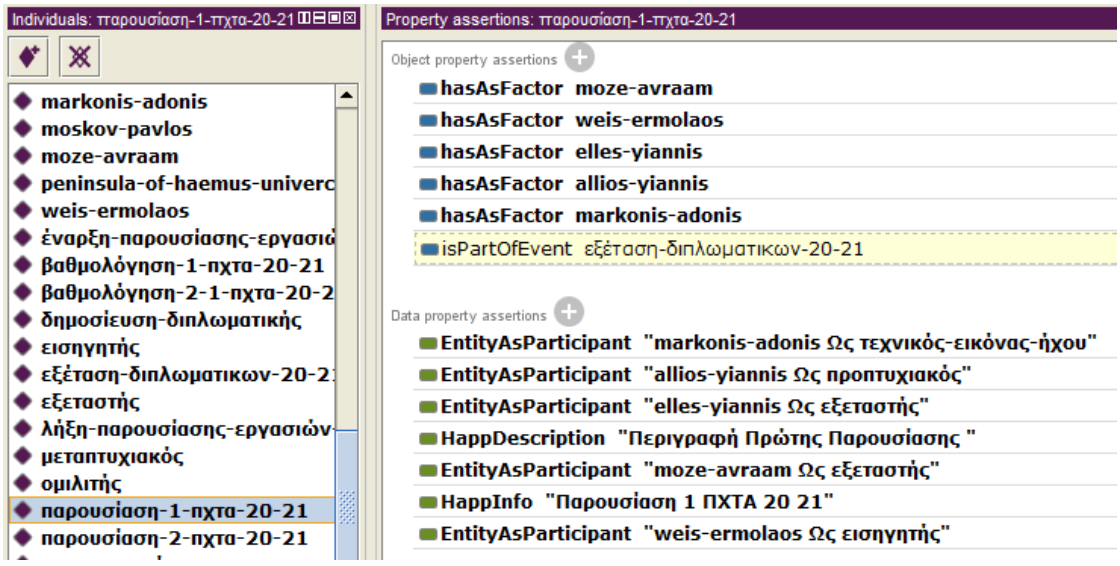
4.2 Έλεγχος στο Protégé

Το αρχείο testMyEvent.ttl άνοιξε χωρίς προβλήματα από το Protégé και ξεκίνησε ο Reasoner HermiT 1.3.8 χωρίς κανένα πρόβλημα.



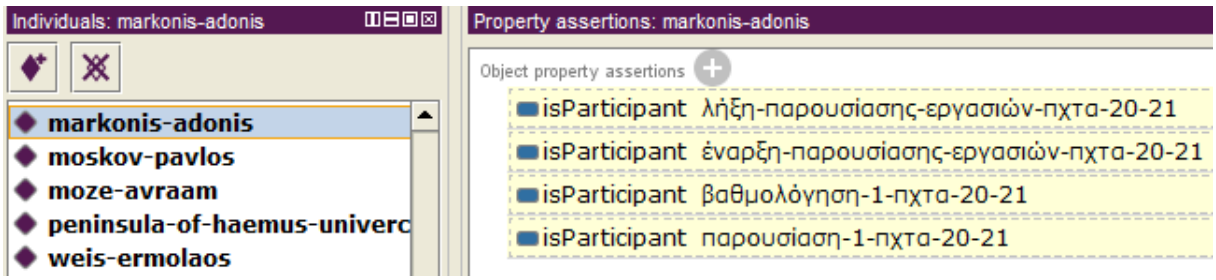
Εικόνα 4.1: Protégé Test Event Individual

Στην *Εικόνα 4.1* εμφανίζονται τα αναμενόμενα αποτελέσματα για την δοκιμαστική εκδήλωση που καταχωρήθηκε στο Κεφάλαιο 3.6.



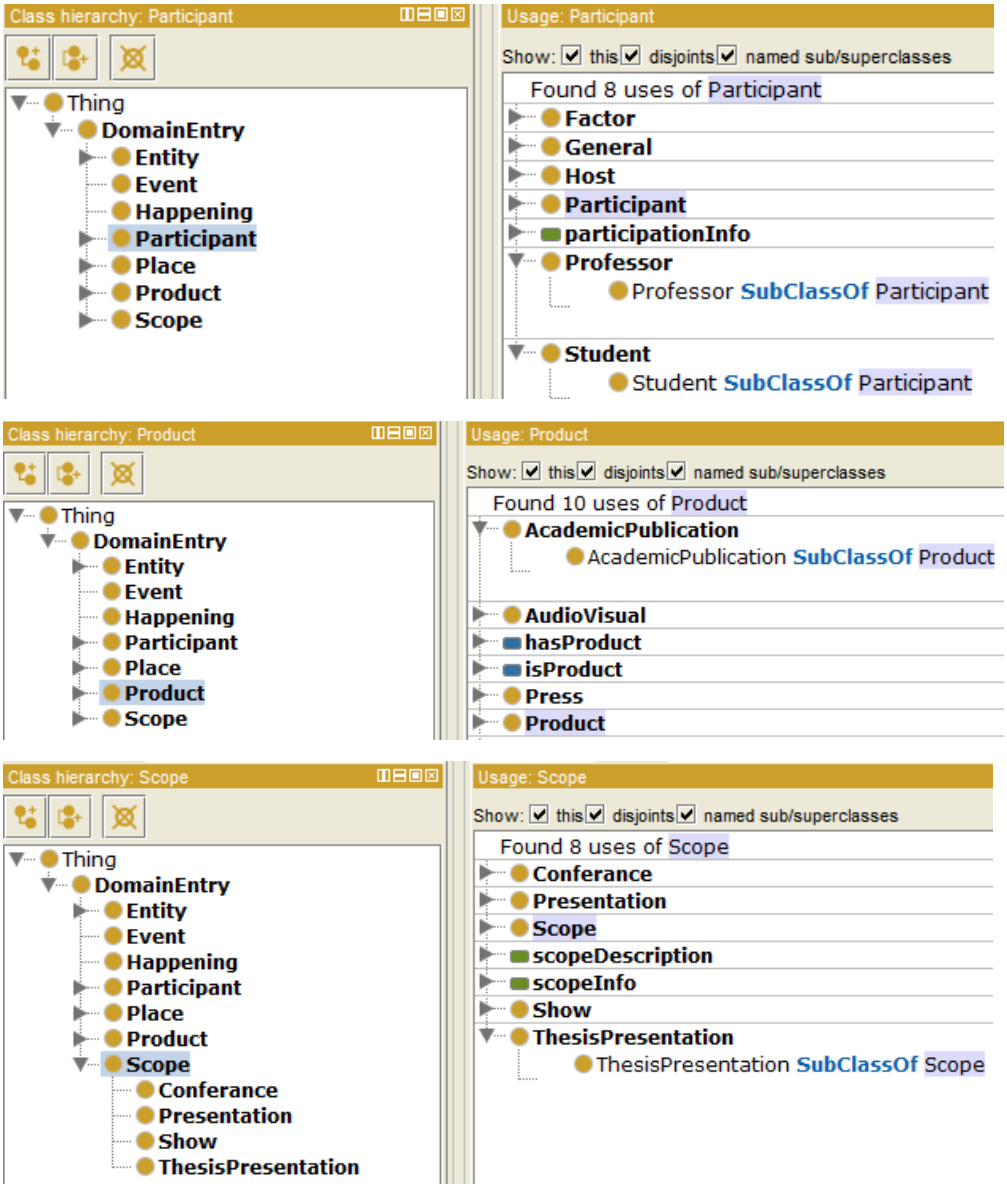
Εικόνα 4.2: Protégé Test Happening Individual

Στην *Εικόνα 4.2* εμφανίζονται τα αναμενόμενα αποτελέσματα για το δοκιμαστικό Happening και παρατηρείται και το συμπέρασμα του Reasoner στο isPartOfEvent.



Εικόνα 4.3: Protégé Test Entity Individual

Στην *Εικόνα 4.3* εμφανίζεται το συμπέρασμα του Reasoner στο isParticipant, η πληροφορία αυτή δεν καταχωρήθηκε αλλά προκύπτει από την αντίστροφη της hasAsFactor του Happening.



Εικόνα 4.4: Protégé Κλάσεις δηλωμένες από χρήστη

Στην *Εικόνα 4.4* εμφανίζονται οι κλάσεις που δηλώθηκαν από τον χρήστη με εντελώς φυσιολογική παρουσία στο Protégé.

4.3 Προβλήματα

Διαπιστώθηκε ότι χρειάζεται η προσθήκη πεδίων ώρας στην κλάση των συμβάντων, αλλά ταυτόχρονα και δημιουργία σύνθετου πεδίου, με τα πεδία αυτά, σε σχέση με την εκδήλωση.

Έγινε ανεπιτυχώς προσπάθεια δημιουργίας διαδικτυακού applet με χρήση Apache Tomcat με σκοπό την εξυπηρέτηση ενός Fuseki server. Η περεταίρω μελέτη της ρύθμισης των διαδικτυακών πόρων ενός server θεωρήθηκε ότι ξεφεύγει από το αντικείμενο αυτής της ΔΕ και εγκαταλείφτηκε.

Έγινε προσπάθεια συνεργασίας της βιβλιοθήκης Jena με την PHP η οποία δεν απέδωσε αναμενόμενα αποτελέσματα και επιλέχθηκε να μην παρουσιαστεί στην ΔΕ. Κατά την έρευνα βρέθηκαν βιβλιοθήκες PHP με σημασιολογική λειτουργία αλλά κρίθηκαν μη επαρκείς και μη ολοκληρωμένες.

Πρόσβαση στο Virtuoso endpoint μέσω PHP είναι δυνατή αλλά περιορίζεται σε απλές POST/GET συναλλαγές που σε περίπτωση (όπως σε αυτή την ΔΕ) που χρειάζεται κάτι περισσότερο από απλή εμφάνιση πινάκων, δεν επαρκούν. Για iteration από κόμβο σε κόμβο με σκοπό την συμπλήρωση στοιχείων HTML απαιτείται η χρήση ή η δημιουργία σημασιολογικής βιβλιοθήκης αυτού του σκοπού.

4.4 Μελλοντική υλοποίηση

Η διαδικτυακή διεπαφή που θα παρουσιάζει τα περιεχόμενα της οντολογίας μέσω endpoint, για το ΠΣ βρίσκεται σε διαδικασία δημιουργίας. Θα ολοκληρωθεί με την διαδικασία update μέσω endpoint στο άμεσο μέλλον.

4.5 Στόχος επόμενης σχετικής έρευνας

Τίθεται σαν στόχος η ανάπτυξη μιας «ελαφριάς» βιβλιοθήκης PHP με την λειτουργικότητα που απαιτεί το ΠΣ που αναπτύχθηκε σε αυτή την ΔΕ. Επιδιωκόμενο αποτέλεσμα θα έχει την αυτοματοποιημένη διερμήνευση της Turtle σύνταξης, για την δημιουργία List και iterator συμβατούς με την λειτουργικότητα του DOM, για την γρήγορη δημιουργία HTML περιεχομένου.

Δεν θα βασιστεί καθόλου αυτή η βιβλιοθήκη σε υπάρχουσες Οντολογικές βιβλιοθήκες αλλά μόνο στις ισχυρές δυνατότητες βιβλιοθηκών PHP για την επεξεργασία κειμένου.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Portal for the Greek language [Online]. Available:

https://www.greek-language.gr/greekLang/modern_greek/tools/lexica/triantafyllides/

[2] Google Dictionary [Online]. Available:

https://www.google.com/search?q=event&oq=event+&aqs.

[3] Online Etymology Dictionary [Online]. Available:

https://www.etymonline.com/word/event

[4] Μαματσή, Ά. (2014). Η Διοργάνωση των Εκδηλώσεων σε Ξενοδοχειακές Μονάδες, η περίπτωση του Νομού Χανίων. Ανώτατο Τεχνολογικό Ίδρυμα Κρήτης, Σχολή Διοίκησης και Οικονομίας, Τμήμα Διοίκησης Επιχειρήσεων, Πρόγραμμα Σπουδών στη Διοίκηση Τουριστικών Επιχειρήσεων.

[5] Κραβαρίτης Κώστας 1992, Επαγγελματικός Τουρισμός Συνεδρίων, Κινήτρων Εκθέσεων, Αθήνα: Interbooks

[6] What is UML [Online]. Available: https://www.visual-paradigm.com/

[7] KDE Community [Online]. Available: https://kde.org/

[8] Umbrello documentation [Online]. Available: https://umbrello.kde.org/documentation.php

[9] Programming Languages for the Semantic Web [Online]. Available:

https://www.w3.org/2001/sw/wiki/Category:Programming_Language

[10] Apache Jena [Online]. Available: https://www.w3.org/2001/sw/wiki/Apache_Jena

[11] Chan, Rosalie (January 22, 2019). Business Insider.

[12] "Write once, run anywhere?". Computer Weekly. 2 Μαΐου 2002.

[13] TIOBE Index "TIOBE Index for May 2021 | TIOBE - The Software Quality Company"

[14] JVM Languages [Online]. Available: https://en.wikipedia.org/wiki/List_of_JVM_languages 21 Μαΐου 2021

[15] "Protégé Community". protege.stanford.edu. 23 May 2020.

[16] "Extensible Markup Language (XML) 1.0"[Online]. Available: www.w3.org.

[17] "XML 1.0 Specification". World Wide Web Consortium [Online]. Available: w3.org

[18] "Extensible Markup Language (XML) 1.1 (Second Edition) , Rationale and list of changes for XML 1.1". [Online]. Available: W3.org

[19] Introduction to DTD. [Online]. Available: https://www.w3schools.com/xml/xml_dtd_intro.asp

[20] XML and Semantic Web W3C Standards Timeline-History, bikakis 2016

[21] Brickley, Dan; Guha, Ramanathan V.; Layman, Andrew, eds. (1998-04-09). "Resource Description Framework (RDF) Schemas". W3C. W3C Working Draft. RDF Schema Working Group

[22] Brickley, Dan; Guha, Ramanathan V., eds. (2014-02-25). "RDF Schema 1.1". W3C. 1.1. RDF Working Group

[23] Berners-Lee, T., Hendler, J., and Lassila, O., The Semantic Web. Scientific American. 2001.

[24] Semantic Web Activity [Online]. Available: https://www.w3.org/2001/sw/

[25] T. B. Passin. Explorer's Guide to the Semantic Web, Manning, 2004.

[26] B. Hoenderboom, P. Liang, A Survey of Semantic Wikis for Requirements

Engineering, University of Groningen, 2009.

[27] Semantic Wb Stack [Online]. Available: https://www.w3.org/2004/Talks/1117-sb-gartnerWS/slide18-0.html

[28] G. Antoniou and F. van Harmelen, A Semantic Web Primer, MIT Press, 2004.[18]

[29]What is an Ontology? Gruber, T. (2001). Stanford University [Online]. Available: https://web.archive.org/web/20100716004426/http:/www-ksl.stanford.edu/kst/what-is-an-ontology.html

[30] Lacy, Lee W. (2005). "Chapter 10". OWL: Representing Information Using the Web Ontology Language. Victoria, BC: Trafford Publishing. ISBN 978-1-4120-3448-7

[31]Horrocks, I. et al.: DAML+OIL. http://www.daml.org/2001/03/daml+oilindex.html (2001)

[32] Dean, M., Schreiber, G.: OWL Web Ontology Language Reference. W3C Recommendation http://www.w3.org/TR/owl-ref/ (2004)

[33] "Eleven SPARQL 1.1 Specifications are W3C Recommendations". w3.org. 21 March 2013

[34] "RDF 1.1 Turtle - Terse RDF Triple LanguageTurtle". World Wide Web Consortium (W3C)

[35] System documentation for RDF4J. The SAIL API

[36] Pellet Reasoner [Online]. Available: https://www.w3.org/2001/sw/wiki/Pellet

[37] Jena : a description of the main Java packages [Online]. Available:

https://jena.apache.org/documentation/rdf/index.html

[38] ARQ - Application API [Online]. Available:

https://jena.apache.org/documentation/query/app_api.html

[39] OpenJFX [Online]. Available: https://wiki.openjdk.java.net/display/OpenJFX/Main

[40] Jena Model [Online]. Available:

https://jena.apache.org/documentation/javadoc/jena/org/apache/jena/rdf/model/

[41]~Okeanos [Online]. Available: https://cyclades.okeanos.grnet.gr/ui/

[42] How to Install Virtuoso Open Source (VOS) on Ubuntu Linux [Online].

Available: http://vos.openlinksw.com/owiki/wiki/VOS/VOSUbuntuNotes

[43] Oracle File Writer [Online]. Available:

https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/io/FileWriter.html

[44]Jena Query [Online]. Available:

https://jena.apache.org/documentation/javadoc/arq/org/apache/jena/query/Query.html

[45] Jena QueryExecution [Online]. Available:
https://jena.apache.org/documentation/javadoc/arq/org/apache/jena/query/QueryExecution.html
[46] Jena ResultSet [Online]. Available:
https://jena.apache.org/documentation/javadoc/arq/org/apache/jena/query/ResultSet.html
[47] Jena QuerySolution [Online]. Available:
https://jena.apache.org/documentation/javadoc/arq/org/apache/jena/query/QuerySolution.html
[48] Jena UpdateAction [Online]. Available:
https://jena.apache.org/documentation/javadoc/arq/org/apache/jena/update/UpdateAction.html

ΠΑΡΑΡΤΗΜΑ Α Κώδικας Turtle Οντολογίας

```
@prefix : <http://www.allios.dx.am#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix base: <http://www.allios.dx.am#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.allios.dx.am> .
<http://www.allios.dx.am> rdf:type owl:Ontology ;
                          owl:versionIRI <http://www.allios.dx.am> .
#################################################################
#
#    Annotation properties
#
#################################################################
###  http://www.allios.dx.am#hasAsFactor
:hasAsFactor rdf:type owl:AnnotationProperty .
#################################################################
#
#    Object Properties
#
#################################################################
###  http://www.allios.dx.am#hasAsFactor
:hasAsFactor rdf:type owl:ObjectProperty ;
             rdfs:range :Entity ;
             rdfs:domain :Happening ;
             owl:inverseOf :isParticipant .
###  http://www.allios.dx.am#hasHappening
:hasHappening rdf:type owl:ObjectProperty ;
              rdfs:domain :Event ;
              owl:inverseOf :isPartOfEvent ;
```

```
            rdfs:range [ rdf:type owl:Restriction ;
                         owl:onProperty :hasHappening ;
                         owl:someValuesFrom :Happening
                       ] .
###  http://www.allios.dx.am#hasHost
:hasHost rdf:type owl:ObjectProperty ;
        rdfs:domain :Event ;
        rdfs:range [ rdf:type owl:Restriction ;
                     owl:onProperty :hasHost ;
                     owl:onClass :Entity ;
                     owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
                   ] .

###  http://www.allios.dx.am#hasPlace
:hasPlace rdf:type owl:ObjectProperty ;
         rdfs:domain :Event ;
         rdfs:range :Place ,
                    [ rdf:type owl:Restriction ;
                      owl:onProperty :hasPlace ;
                      owl:onClass :Place ;
                      owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
                    ] .
###  http://www.allios.dx.am#hasProduct
:hasProduct rdf:type owl:ObjectProperty ;
           rdfs:domain :Event ;
           rdfs:range :Product ;
           owl:inverseOf :isProduct ;
           rdfs:subPropertyOf owl:topObjectProperty .

###  http://www.allios.dx.am#isPartOfEvent
:isPartOfEvent rdf:type owl:ObjectProperty ;
              rdfs:comment "Ενα Happening ανήκει σε Ενα EventnTo Event έχει πολλά Happening" ;
              rdfs:domain :Happening ;
```

```
        rdfs:comment :isPartOfEvent ;
        rdfs:range [ rdf:type owl:Restriction ;
                     owl:onProperty :isPartOfEvent ;
                     owl:onClass :Event ;
                     owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
                   ] .
###  http://www.allios.dx.am#isParticipant
:isParticipant rdf:type owl:ObjectProperty ;
        rdfs:domain :Entity ;
        rdfs:range :Happening .

###  http://www.allios.dx.am#isProduct
:isProduct rdf:type owl:ObjectProperty ;
        rdfs:domain :Product ;
        rdfs:subPropertyOf owl:topObjectProperty ;
        rdfs:range [ rdf:type owl:Restriction ;
                     owl:onProperty :isProduct ;
                     owl:onClass :Event ;
                     owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
                   ] .
#################################################################
#
#    Data properties
#
#################################################################
###  http://www.allios.dx.am#EntityAsParticipant
:EntityAsParticipant rdf:type owl:DatatypeProperty ;
                     rdfs:comment "O EntityIndividual είναι ένα ParticipationIndividualno διαχωρισμός γίνεται με το \" ως \"" ;
                     rdfs:domain :Happening ;
                     rdfs:subPropertyOf :HappeningData ;
                     rdfs:range xsd:string .
```

```
###  http://www.allios.dx.am#EntityDescription
:EntityDescription rdf:type owl:DatatypeProperty ;
                   rdfs:domain :Entity ;
                   rdfs:subPropertyOf :entityData ;
                   rdfs:range xsd:string .
###  http://www.allios.dx.am#EntityLegalName
:EntityLegalName rdf:type owl:DatatypeProperty ;
                 rdfs:domain :Legal ;
                 rdfs:subPropertyOf :entityData ;
                 rdfs:range xsd:string .
###  http://www.allios.dx.am#EntityPhisicalName
:EntityPhisicalName rdf:type owl:DatatypeProperty ;
                    rdfs:domain :Physical ;
                    rdfs:subPropertyOf :entityData ;
                    rdfs:range xsd:string .
###  http://www.allios.dx.am#EventBeginDate
:EventBeginDate rdf:type owl:DatatypeProperty ;
                rdfs:domain :Event ;
                rdfs:subPropertyOf :eventData ;
                rdfs:range xsd:string .
###  http://www.allios.dx.am#EventDescription
:EventDescription rdf:type owl:DatatypeProperty ;
                  rdfs:domain :Event ;
                  rdfs:subPropertyOf :eventData ;
                  rdfs:range xsd:string .
###  http://www.allios.dx.am#EventEndDate
:EventEndDate rdf:type owl:DatatypeProperty ;
              rdfs:domain :Event ;
              rdfs:subPropertyOf :eventData ;
              rdfs:range xsd:string .
###  http://www.allios.dx.am#EventTitle
:EventTitle rdf:type owl:DatatypeProperty ;
            rdfs:domain :Event ;
```

```
        rdfs:subPropertyOf :eventData ;
        rdfs:range xsd:string .
###  http://www.allios.dx.am#HappDescription
:HappDescription rdf:type owl:DatatypeProperty ;
            rdfs:domain :Happening ;
            rdfs:subPropertyOf :HappeningData ;
            rdfs:range xsd:string .
###  http://www.allios.dx.am#HappInfo
:HappInfo rdf:type owl:DatatypeProperty ;
        rdfs:domain :Happening ;
        rdfs:subPropertyOf :HappeningData ;
        rdfs:range xsd:string .

###  http://www.allios.dx.am#HappeningData
:HappeningData rdf:type owl:DatatypeProperty .

###  http://www.allios.dx.am#PlaceAddress
:PlaceAddress rdf:type owl:DatatypeProperty ;
            rdfs:domain :Actual ;
            rdfs:subPropertyOf :placeData ;
            rdfs:range xsd:string .

###  http://www.allios.dx.am#PlaceBuilding
:PlaceBuilding rdf:type owl:DatatypeProperty ;
            rdfs:domain :Actual ;
            rdfs:subPropertyOf :placeData ;
            rdfs:range xsd:string .

###  http://www.allios.dx.am#PlaceCity
:PlaceCity rdf:type owl:DatatypeProperty ;
        rdfs:domain :Actual ;
        rdfs:subPropertyOf :placeData ;
        rdfs:range xsd:string .
```

```
###  http://www.allios.dx.am#PlaceCountry
:PlaceCountry rdf:type owl:DatatypeProperty ;
              rdfs:domain :Actual ;
              rdfs:subPropertyOf :placeData ;
              rdfs:range xsd:string .

###  http://www.allios.dx.am#PlaceDiscription
:PlaceDiscription rdf:type owl:DatatypeProperty ;
                  rdfs:domain :Place ;
                  rdfs:subPropertyOf :placeData ;
                  rdfs:range xsd:string .

###  http://www.allios.dx.am#PlacePlatform
:PlacePlatform rdf:type owl:DatatypeProperty ;
               rdfs:domain :Virtual ;
               rdfs:subPropertyOf :placeData ;
               rdfs:range xsd:string .

###  http://www.allios.dx.am#PlaceWebAddress
:PlaceWebAddress rdf:type owl:DatatypeProperty ;
                 rdfs:domain :Virtual ;
                 rdfs:subPropertyOf :placeData ;
                 rdfs:range xsd:string .

###  http://www.allios.dx.am#ProductData
:ProductData rdf:type owl:DatatypeProperty .

###  http://www.allios.dx.am#ProductDescription
:ProductDescription rdf:type owl:DatatypeProperty ;
                    rdfs:domain :Product ;
                    rdfs:subPropertyOf :ProductData ;
                    rdfs:range xsd:string .
```

```
###  http://www.allios.dx.am#ProductInfo
:ProductInfo rdf:type owl:DatatypeProperty ;
             rdfs:domain :Product ;
             rdfs:subPropertyOf :ProductData ;
             rdfs:range xsd:string .

###  http://www.allios.dx.am#entityData
:entityData rdf:type owl:DatatypeProperty ;
            rdfs:domain :Entity .

###  http://www.allios.dx.am#eventData
:eventData rdf:type owl:DatatypeProperty .

###  http://www.allios.dx.am#participationInfo
:participationInfo rdf:type owl:DatatypeProperty ;
                   rdfs:domain :Participant ;
                   rdfs:range xsd:string .

###  http://www.allios.dx.am#placeData
:placeData rdf:type owl:DatatypeProperty ;
           rdfs:domain :Place ;
           rdfs:range xsd:string .

###  http://www.allios.dx.am#scopeData
:scopeData rdf:type owl:DatatypeProperty .

###  http://www.allios.dx.am#scopeDescription
:scopeDescription rdf:type owl:DatatypeProperty ;
                  rdfs:domain :Scope ;
                  rdfs:subPropertyOf :scopeData ;
                  rdfs:range xsd:string .
```

```
###  http://www.allios.dx.am#scopeInfo
:scopeInfo rdf:type owl:DatatypeProperty ;
          rdfs:domain :Scope ;
          rdfs:subPropertyOf :scopeData ;
          rdfs:range xsd:string .

#################################################################
#
#    Classes
#
#################################################################
###  http://www.allios.dx.am#Actual
:Actual rdf:type owl:Class ;
        rdfs:subClassOf :Place .

###  http://www.allios.dx.am#AudioVisual
:AudioVisual rdf:type owl:Class ;
            rdfs:subClassOf :Product .

###  http://www.allios.dx.am#Awards
:Awards rdf:type owl:Class ;
        rdfs:subClassOf :Scope ;
        rdfs:comment "Βράβευση" .

###  http://www.allios.dx.am#Conference
:Conference rdf:type owl:Class ;
            rdfs:subClassOf :Scope ;
            rdfs:comment "Συνέδριο" .

###  http://www.allios.dx.am#DomainEntry
:DomainEntry rdf:type owl:Class .

###  http://www.allios.dx.am#Entity
```

```
:Entity rdf:type owl:Class ;
       rdfs:subClassOf :DomainEntry .

###  http://www.allios.dx.am#Event
:Event rdf:type owl:Class ;
      rdfs:subClassOf :DomainEntry ,
                      [ rdf:type owl:Restriction ;
                        owl:onProperty :hasHappening ;
                        owl:someValuesFrom :Happening
                      ] .

###  http://www.allios.dx.am#Factor
:Factor rdf:type owl:Class ;
       rdfs:subClassOf :Participant ;
       rdfs:comment "Παράγοντας Εκδήλωσης (Event only)" .

###  http://www.allios.dx.am#General
:General rdf:type owl:Class ;
        rdfs:subClassOf :Participant ;
        rdfs:comment "Συμμετέχουν στα Happening (Happening only)" .

###  http://www.allios.dx.am#Happening
:Happening rdf:type owl:Class ;
         rdfs:subClassOf :DomainEntry ,
                         [ rdf:type owl:Restriction ;
                           owl:onProperty :isPartOfEvent ;
                           owl:onClass :Event ;
                           owl:qualifiedCardinality "1"^^xsd:nonNegativeInteger
                         ] .

###  http://www.allios.dx.am#Host
:Host rdf:type owl:Class ;
     rdfs:subClassOf :Participant ;
```

```
    rdfs:comment "Οικοδεσπότης Εκδήλωσης (Event only)" .

###  http://www.allios.dx.am#Legal
:Legal rdf:type owl:Class ;
      rdfs:subClassOf :Entity ;
      rdfs:comment "Μη Φυσικό ΠρόσωποπονΕταιρεία, οργανισμός, NGO κ.τ.λ" .

###  http://www.allios.dx.am#Participant
:Participant rdf:type owl:Class ;
           rdfs:subClassOf :DomainEntry .

###  http://www.allios.dx.am#Physical
:Physical rdf:type owl:Class ;
         rdfs:subClassOf :Entity ;
         rdfs:comment "Φυσικό Πρόσωπο" .

###  http://www.allios.dx.am#Place
:Place rdf:type owl:Class ;
      rdfs:subClassOf :DomainEntry .

###  http://www.allios.dx.am#Presentation
:Presentation rdf:type owl:Class ;
            rdfs:subClassOf :Scope ;
            rdfs:comment "Παρουσίαση (Προϊόντος, βιβλίου, Εργασίας)" .

###  http://www.allios.dx.am#Press
:Press rdf:type owl:Class ;
      rdfs:subClassOf :Product .

###  http://www.allios.dx.am#Product
:Product rdf:type owl:Class ;
        rdfs:subClassOf :DomainEntry .
```

```
###  http://www.allios.dx.am#Publication
:Publication rdf:type owl:Class ;
             rdfs:subClassOf :Product .

###  http://www.allios.dx.am#Scope
:Scope rdf:type owl:Class ;
       rdfs:subClassOf :DomainEntry .

###  http://www.allios.dx.am#Virtual
:Virtual rdf:type owl:Class ;
         rdfs:subClassOf :Place .

###  Generated by the OWL API (version 3.4.2) http://owlapi.sourceforge.net
```

ΠΑΡΑΡΤΗΜΑ Β Κώδικας Turtle Individuals

```
#################################################################
#
#    Individuals
#
#################################################################
###  http://www.allios.dx.am#IND-ENT-LEGAL-01
:IND-ENT-LEGAL-01 rdf:type :Legal ,
                          owl:NamedIndividual ;
                  :EntityDescription "περιγραφή entity 01" ;
                  :EntityLegalName "επωνυμία Νομικής Entity 01" .

###  http://www.allios.dx.am#IND-ENT-LEGAL-02
:IND-ENT-LEGAL-02 rdf:type :Legal ,
                          owl:NamedIndividual ;
                  :EntityDescription "περιγραφή entity 02" ;
                  :EntityLegalName "επωνυμία Νομικής Entity 02" .

###  http://www.allios.dx.am#IND-ENT-LEGAL-03
:IND-ENT-LEGAL-03 rdf:type :Legal ,
                          owl:NamedIndividual ;
                  :EntityLegalName "επωνυμία Νομικής Entity 03" ;
                  :EntityDescription "περιγραφή entity 03" .

###  http://www.allios.dx.am#IND-ENT-PHY-01
:IND-ENT-PHY-01 rdf:type :Physical ,
                         owl:NamedIndividual ;
                 :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 1" ;
                 :EntityDescription "Περιγραφή φυσικής Οντότητας 1" ;
                 :isParticipant :IND-HAPP-01 ,
                                :IND-HAPP-05 .
```

```
###  http://www.allios.dx.am#IND-ENT-PHY-02
:IND-ENT-PHY-02 rdf:type :Physical ,
                    owl:NamedIndividual ;
            :EntityDescription "Περιγραφή φυσικής Οντότητας 2" ;
            :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 2" ;

            :isParticipant :IND-HAPP-01 .

###  http://www.allios.dx.am#IND-ENT-PHY-03
:IND-ENT-PHY-03 rdf:type :Physical ,
                    owl:NamedIndividual ;
            :EntityDescription "Περιγραφή φυσικής Οντότητας 3" ;
            :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 3" ;
            :isParticipant :IND-HAPP-05 .

###  http://www.allios.dx.am#IND-ENT-PHY-04
:IND-ENT-PHY-04 rdf:type :Physical ,
                    owl:NamedIndividual ;
            :EntityDescription "Περιγραφή φυσικής Οντότητας 4" ;
            :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 4" ;
            :isParticipant :IND-HAPP-01 ,
                        :IND-HAPP-06 .

###  http://www.allios.dx.am#IND-ENT-PHY-05
:IND-ENT-PHY-05 rdf:type :Physical ,
                    owl:NamedIndividual ;
            :EntityDescription "Περιγραφή φυσικής Οντότητας 5" ;
            :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 5" ;
            :isParticipant :IND-HAPP-03 .

###  http://www.allios.dx.am#IND-ENT-PHY-06
:IND-ENT-PHY-06 rdf:type :Physical ,
                    owl:NamedIndividual ;
```

```
        :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 6" ;
        :EntityDescription "Περιγραφή φυσικής Οντότητας 6" ;
        :isParticipant :IND-HAPP-04 .

###  http://www.allios.dx.am#IND-ENT-PHY-07
:IND-ENT-PHY-07 rdf:type :Physical ,
                 owl:NamedIndividual ;
        :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 7" ;
        :EntityDescription "Περιγραφή φυσικής Οντότητας 7" ;
        :isParticipant :IND-HAPP-06 .

###  http://www.allios.dx.am#IND-ENT-PHY-08
:IND-ENT-PHY-08 rdf:type :Physical ,
                 owl:NamedIndividual ;
        :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 8" ;
        :EntityDescription "Περιγραφή φυσικής Οντότητας 8" ;
        :isParticipant :IND-HAPP-04 .

###  http://www.allios.dx.am#IND-ENT-PHY-09
:IND-ENT-PHY-09 rdf:type :Physical ,
                 owl:NamedIndividual ;
        :EntityDescription "Περιγραφή φυσικής Οντότητας 9" ;
        :EntityPhisicalName "Ονοματεπώνυμο φυσικής οντότητας 9" ;
        :isParticipant :IND-HAPP-01 .

###  http://www.allios.dx.am#IND-EVENT-01
:IND-EVENT-01 rdf:type :Event ,
                :Show ,
                owl:NamedIndividual ;
        :scopeDescription "Περιγραφή Σκοπού εκδήλωσης 1"^^xsd:string ;
        :scopeInfo "Πληροφορίες Σκοπού εκδήλωσης 1"^^xsd:string ;
        :EventDescription "Περιγραφή Εκδήλωσης 1" ;
        :EventEndDate "2021-04-01" ;
```

```
        :EventTitle "Τίτλος Εκδήλωσης 1" ;
        :EventBeginDate "2021-04-01" ;
        :hasHost :IND-ENT-LEGAL-01 ;
        :hasHappening :IND-HAPP-01 ,
                      :IND-HAPP-02 ,
                      :IND-HAPP-03 ;
        :hasPlace :IND-PLACE-ACTUAL ;
        :hasProduct :IND-PRODUCT-01 ,
                    :IND-PRODUCT-02 .

###  http://www.allios.dx.am#IND-EVENT-02
:IND-EVENT-02 rdf:type :Event ,
                       :Presentation ,
                       owl:NamedIndividual ;
        :scopeDescription "Πληροφορίες Σκοπού εκδήλωσης 2"^^xsd:string ;
        :EventTitle "Τίτλος Εκδήλωσης 2" ;
        :EventDescription "Περιγραφή Εκδήλωσης 2" ;
        :EventBeginDate "2021-04-15" ;
        :scopeDescription "Περιγραφή Σκοπού Εκδήλωσης 2" ;
        :EventEndDate "2021-04-15" ;
        :hasHost :IND-ENT-PHY-01 ;
        :hasHappening :IND-HAPP-04 ,
                      :IND-HAPP-05 ,
                      :IND-HAPP-06 ;
        :hasPlace :IND-PLACE-ACTUAL ;
        :hasProduct :IND-PRODUCT-03 ,
                    :IND-PRODUCT-04 .

###  http://www.allios.dx.am#IND-HAPP-01
:IND-HAPP-01 rdf:type :Happening ,
                      owl:NamedIndividual ;
       :EntityAsParticipant "IND-ENT-PHY-01 ως IND-PAR-GENERAL-03"^^xsd:string ,
                            "IND-ENT-PHY-02 ως IND-PAR-GENERAL-02"^^xsd:string ,
```

```
                    "IND-ENT-PHY-04 ως IND-PAR-GENERAL-01"^^xsd:string ,
                     "IND-ENT-PHY-09 ως IND-PAR-FACTOR-01"^^xsd:string ;
        :HappDescription "Περιγραφή Happening 1" ;
        :HappInfo "Πληροφορίες Happening 1" ;
        :hasAsFactor :IND-ENT-PHY-02 ;
        :isPartOfEvent :IND-EVENT-01 .

###  http://www.allios.dx.am#IND-HAPP-02
:IND-HAPP-02 rdf:type :Happening ,
                owl:NamedIndividual ;
       :EntityAsParticipant "IND-ENT-PHY-03 ως IND-PAR-FACTOR-01"^^xsd:string ,
                    "IND-ENT-PHY-05 ως IND-PAR-GENERAL-01"^^xsd:string ,
                    "IND-ENT-PHY-06 ως IND-PAR-GENERAL-01"^^xsd:string ;
        :HappInfo "Πληροφορίες Happening 2" ;
        :HappDescription "Περιγραφή Happening 2" ;
        :hasAsFactor :IND-ENT-PHY-03 ,
                :IND-ENT-PHY-05 ,
                :IND-ENT-PHY-06 ;
        :isPartOfEvent :IND-EVENT-01 .

###  http://www.allios.dx.am#IND-HAPP-03
:IND-HAPP-03 rdf:type :Happening ,
                owl:NamedIndividual ;
       :EntityAsParticipant "IND-ENT-PHY-05 ως IND-PAR-GENERAL-03"^^xsd:string ;
        :HappInfo "Πληροφορίες Happening 3" ;
        :HappDescription "Περιγραφή Happening 3" ;
        :isPartOfEvent :IND-EVENT-01 .

###  http://www.allios.dx.am#IND-HAPP-04
:IND-HAPP-04 rdf:type :Happening ,
                owl:NamedIndividual ;
       :EntityAsParticipant "IND-ENT-PHY-06 ως IND-PAR-GENERAL-03"^^xsd:string ,
                    "IND-ENT-PHY-08 ως IND-PAR-GENERAL-02"^^xsd:string ;
```

```
        :HappDescription "Περιγραφή Happening 4" ;
        :HappInfo "Πληροφορίες Happening 4" ;
        :isPartOfEvent :IND-EVENT-02 .

###  http://www.allios.dx.am#IND-HAPP-05
:IND-HAPP-05 rdf:type :Happening ,
                owl:NamedIndividual ;
       :EntityAsParticipant "IND-ENT-PHY-01 ως IND-PAR-GENERAL-02"^^xsd:string ,
                    "IND-ENT-PHY-03 ως IND-PAR-GENERAL-02"^^xsd:string ;
        :HappInfo "Πληροφορίες Happening 5" ;
        :HappDescription "Περιγραφή Happening 5" ;
        :isPartOfEvent :IND-EVENT-02 .

###  http://www.allios.dx.am#IND-HAPP-06
:IND-HAPP-06 rdf:type :Happening ,
                owl:NamedIndividual ;
       :EntityAsParticipant "IND-ENT-PHY-04 ως IND-PAR-GENERAL-02"^^xsd:string ,
                    "IND-ENT-PHY-07 ως IND-PAR-GENERAL-02"^^xsd:string ;
        :HappInfo "Πληροφορίες Happening 1" ;
        :HappDescription "Περιγραφή Happening 6" ;
        :isPartOfEvent :IND-EVENT-02 .

###  http://www.allios.dx.am#IND-PAR-FACTOR-01
:IND-PAR-FACTOR-01 rdf:type :Factor ,
                    owl:NamedIndividual ;
              :participationInfo "Catering" .

###  http://www.allios.dx.am#IND-PAR-FACTOR-02
:IND-PAR-FACTOR-02 rdf:type :Factor ,
                    owl:NamedIndividual ;
              :participationInfo "Φύλαξη - Security" .

###  http://www.allios.dx.am#IND-PAR-FACTOR-03
```

```
:IND-PAR-FACTOR-03 rdf:type :Factor ,
                   owl:NamedIndividual ;
          :participationInfo "Ηλεκτρολογικά - Φωτισμός" .

###  http://www.allios.dx.am#IND-PAR-FACTOR-04
:IND-PAR-FACTOR-04 rdf:type :Factor ,
                   owl:NamedIndividual ;
          :participationInfo "Διακόσμηση - Κατασκευές" .

###  http://www.allios.dx.am#IND-PAR-FACTOR-05
:IND-PAR-FACTOR-05 rdf:type :Factor ,
                   owl:NamedIndividual ;
          :participationInfo "Μακιγιάζ - Φροντιστές" .

###  http://www.allios.dx.am#IND-PAR-FACTOR-06
:IND-PAR-FACTOR-06 rdf:type :Factor ,
                   owl:NamedIndividual ;
          :participationInfo "Υποστήριξη Η/Υ - Δικτύου Η/Υ" .

###  http://www.allios.dx.am#IND-PAR-GENERAL-01
:IND-PAR-GENERAL-01 rdf:type :General ,
                    owl:NamedIndividual ;
           :participationInfo "Τιμώμενο Πρόσωπο" .

###  http://www.allios.dx.am#IND-PAR-GENERAL-02
:IND-PAR-GENERAL-02 rdf:type :General ,
                    owl:NamedIndividual ;
           :participationInfo "Ομιλητής" .

###  http://www.allios.dx.am#IND-PAR-GENERAL-03
:IND-PAR-GENERAL-03 rdf:type :General ,
                    owl:NamedIndividual ;
          :participationInfo "Performer Καλλιτέχνης" .
```

```
### http://www.allios.dx.am#IND-PAR-HOST-01
:IND-PAR-HOST-01 rdf:type :Host ,
                 owl:NamedIndividual ;
          :participationInfo "Διοργανωτής Εκδήλωσης" .

### http://www.allios.dx.am#IND-PLACE-ACTUAL
:IND-PLACE-ACTUAL rdf:type :Actual ,
                  owl:NamedIndividual ;
          :PlaceCity "πόλη του Χώρου Εκδήλωσης" ;
          :PlaceWebAddress "www.someplace.com" ;
          :PlaceCountry "Χώρα" ;
          :PlaceAddress "Διεύθυνση Φυσικής Τοποθεσίας" ;
          :PlaceBuilding "Κτήριο ή Όνομα χώρου" ;
          :PlaceDiscription "Περιγραφή Χώρου εκδήλωσης" .

### http://www.allios.dx.am#IND-PLACE-VIRTUAL
:IND-PLACE-VIRTUAL rdf:type :Virtual ,
                   owl:NamedIndividual ;
          :PlaceDiscription "Περιγραφή Χώρου Εκδήλωσης" ;
          :PlacePlatform "Πλατφόρμα  Εικονικού Χώρου" .

### http://www.allios.dx.am#IND-PRODUCT-01
:IND-PRODUCT-01 rdf:type :Press ,
                owl:NamedIndividual ;
          :ProductInfo "Τίτλος Προϊόντος 1" ;
          :ProductDescription "Περιγραφή και Πληροφορίες Παραγόμενου Προϊόντος της
Εκδήλωσης 1" ;
          :isProduct :IND-EVENT-01 .

### http://www.allios.dx.am#IND-PRODUCT-02
:IND-PRODUCT-02 rdf:type :Publication ,
                owl:NamedIndividual ;
```

:ProductInfo "Τίτλος Προϊόντος 2" ;

:ProductDescription "Περιγραφή και Πληροφορίες Παραγόμενου Προϊόντος της Εκδήλωσης 2" ;

:isProduct :IND-EVENT-01 .

http://www.allios.dx.am#IND-PRODUCT-03

:IND-PRODUCT-03 rdf:type :Publication ,

owl:NamedIndividual ;

:ProductInfo "Τίτλος Προϊόντος 3" ;

:ProductDescription "Περιγραφή και Πληροφορίες Παραγόμενου Προϊόντος της Εκδήλωσης 3" ;

:isProduct :IND-EVENT-02 .

http://www.allios.dx.am#IND-PRODUCT-04

:IND-PRODUCT-04 rdf:type :AudioVisual ,

owl:NamedIndividual ;

:ProductInfo "Τίτλος Προϊόντος 4" ;

:ProductDescription "Περιγραφή και Πληροφορίες Παραγόμενου Προϊόντος της Εκδήλωσης 4" ;

:isProduct :IND-EVENT-02 .

Generated by the OWL API (version 3.4.2) http://owlapi.sourceforge.net

ΠΑΡΑΡΤΗΜΑ C Java JavaFX FXML

public class allios

```
{
   /*  Παράδειγμα κλήσης
                Authenticator.setDefault(new allios.MySparqlAuth());
   */
   public static class MySparqlAuth extends Authenticator
   {
      protected PasswordAuthentication getPasswordAuthentication()
      {
      // String username = "***"; String password = "***"; // for my site
         String username = "***"; String password = "***"; // for DIPAE site
         return new PasswordAuthentication(username, password.toCharArray());
      }
   }

   static void Say(String string){System.out.println(string);}

   public static String getFrontFromMixString(String mixedTextField)
   {
      String theEntity =":";
      theEntity += mixedTextField.substring(0, (mixedTextField.indexOf("ως"))-1);
      return theEntity;
   }

     public static String getFrontFromMixString(String mixedTextField, String seperator)
   {
      String theEntity =":";
      theEntity += mixedTextField.substring(0, (mixedTextField.indexOf(seperator))-1);
      return theEntity;
   }
```

```
public static String getTheBacktFromMixString(String mixedTextField)
{
   String theRole =":";
   theRole += mixedTextField.substring(mixedTextField.indexOf("ως"));
   return theRole;
}

public static String getTheBacktFromMixString(String mixedTextField, String seperator)
{
   String theRole =":";
   theRole += mixedTextField.substring(mixedTextField.indexOf(seperator));
   return theRole;
}


static String getMyPrefix()
{
   return "prefix : <http://www.allios.dx.am#> \n" +
          "prefix my: <http://www.allios.dx.am#> \n" +
          "prefix owl: <http://www.w3.org/2002/07/owl#> \n" +
          "prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> \n" +
          "prefix xml: <http://www.w3.org/XML/1998/namespace> \n" +
          "prefix xsd: <http://www.w3.org/2001/XMLSchema#> \n" +
          "prefix base: <http://www.allios.dx.am#> \n" +
          "prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> \n" +
          "";
}

static String getMyPrefix(String url)
{
   return "prefix : <"+ url +"#> \n" +
          "prefix my: <"+ url +"#> \n" +
          "prefix owl: <http://www.w3.org/2002/07/owl#> \n" +
```

```
        "prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> \n" +
        "prefix xml: <http://www.w3.org/XML/1998/namespace> \n" +
        "prefix xsd: <http://www.w3.org/2001/XMLSchema#> \n" +
        "prefix base: <"+ url +"#> \n" +
        "prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> \n" +
        "\n";
}


static String specialCharRemove(String theString)
{
   String legalString="";
   legalString = theString.trim()
   .replaceAll("\\t", "")
   .replaceAll("\\n", "")
   .replaceAll(":", "")
   .replaceAll(",", "")
   .replaceAll(";", "")
   .replaceAll("\\.", "")
   .replaceAll("\\^", "")
   .replaceAll("\"", "")
   .replaceAll("/", "")
   .replaceAll("#", "")
   .replaceAll("<", "")
   .replaceAll(">", "")
   .replaceAll("@", "")
   .replaceAll("\'", "");
   return legalString;
}


static String makeLegal(String theString)
{
```

```
    String legalString="";
    legalString = theString.trim().toLowerCase()
    .replaceAll("\\s", "-")
    .replaceAll(":", "")
    .replaceAll(",", "")
    .replaceAll(";", "")
    .replaceAll("\\.", "")
    .replaceAll("\\^", "")
    .replaceAll("\'", "")
    .replaceAll("\"", "");
    return legalString;
}

static String makeLegalClass(String theString)//π
{
    String legalString="";
    legalString = theString.trim()
    .replaceAll("\\s", "-")
    .replaceAll(":", "")
    .replaceAll(",", "")
    .replaceAll(";", "")
    .replaceAll("\\.", "")
    .replaceAll("\\^", "")
    .replaceAll("\'", "")
    .replaceAll("\"", "");
    return legalString;
}

static String noNewLinesNoTabs(String theString)
{
    return  theString = theString.replaceAll("[\\n\\t]", " ");
}
```

```
static Exception WriteModelToTurtle(Model theModel, String theFile) throws Exception
{
   allios.Say("This is WriteModelToTurtle");
   Exception myEx = null;
   try
   {
      FileWriter out = new FileWriter( theFile);
      theModel.write(out, "TURTLE");
      out.close();
      theModel.close();
   }
   catch(IOException closeException){myEx = closeException;}
return     myEx;
}


static Model getModelFromTTLFile(String theFile)//p
{
   Model theModel = null;
   try
   {
      File f = new File(theFile);
      String path = f.getAbsolutePath();
      theModel =
       ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM).read(path);
   }
   catch(Exception e){System.out.println(e.toString()); System.out.println("this is catch");}
   return theModel;
}

static Model getModelFromTTLFile()
{
   Model theModel = null;
```

```
    try
    {
        File f = new File("Ontology/allios.ttl");
        String path = f.getAbsolutePath();
        theModel =
         ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM).read(path);
    }
    catch(Exception e){System.out.println(e.toString()); System.out.println("this is catch");}
    return theModel;
}


static Exception WriteModelToRDF(Model theModel, String theFile) throws Exception
{
    Exception myEx = null;
    try
    {
        FileWriter out = new FileWriter( theFile);
        theModel.write(out, "RDF/XML-ABBREV");
        out.close();
    }
    catch(IOException closeException){myEx = closeException;}
return     myEx;
}

static ArrayList getAllUserNames(Model theModel)
{
   ArrayList theList;
   try
   {
      String theQuery = allios.getMyPrefix()+
      "SELECT   ?value "+
      "WHERE {"+
```

```
        "{ ?b rdf:type owl:Class. "+
        "?value rdf:type ?b.  } "+
        "UNION {?value rdf:type owl:DatatypeProperty.}"+
        "UNION {?value rdf:type owl:ObjectProperty.} "+
        " } "+
        " group by ?value "+
        " order by ?value ";
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
        theList = allios.getArrayLisFromQueryResults(results);

    }
    catch(Exception e)
    {
        allios.Say(e.toString());
        theList = null;
    }
    return theList;
}

static ArrayList getValuesOfPropertiresOfIndividual(Model theModel, String individual)
{
     ArrayList theList;
     try
     {
          String theQuery = allios.getMyPrefix()+
              "SELECT   ?value "+
              "WHERE {"+
           ":"+individual+"   rdf:type owl:NamedIndividual . "+
                      ":"+individual+" ?atribute ?value. "+
              " minus{ ?individual rdf:type ?value.} "+
              "} "+
```

```
            "group by   ?value "+
            "order by ?value";
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    theList = allios.getArrayLisFromQueryResults(results);
    return theList;
    }
    catch(Exception e){allios.Say(e.toString());theList = null;}
    return theList;
}

static ArrayList getValuesOfPropertiresOfClass(Model theModel, String theClass)
{
    ArrayList theList;
    try
    {
//      allios.Say(theClass);
        String theQuery = allios.getMyPrefix()+
            "SELECT   ?value "+
            "WHERE {"+
         ":"+theClass+"   rdf:type owl:Class . "+
                  ":"+theClass+" ?atribute ?value. "+
            " minus{ ?a rdf:type ?value.} "+
            " minus{ ?value rdf:type owl:Class.} "+
            "} "+
            "group by   ?value "+
            "order by ?value";
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    theList = allios.getArrayLisFromQueryResults(results);
    return theList;
```

```
    }
    catch(Exception e){allios.Say(e.toString());theList = null;}
    return theList;
}


static ArrayList  getMembersOfSuperClass(Model theModel, String superClass)
{
    String theQuery = allios.getMyPrefix()+
                    "SELECT ?a  WHERE { ?a rdfs:subClassOf :"+superClass+" }";
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    ArrayList<String> theList = allios.getArrayLisFromQueryResults(results);
    return theList;
}

static ObservableList<String>  getMembersOfClass(Model theModel, String className)
{
    String theQuery = allios.getMyPrefix()+
                    "SELECT ?a  WHERE { ?a rdf:type :"+className+" }";
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    ObservableList<String> theList = allios.getObervableLisFromQueryResults(results);
    return theList;
}


static ArrayList  getAllIndividualsOfClass(Model theModel, String className)
{
    String theQuery = allios.getMyPrefix()+
                    "SELECT ?a  WHERE { ?a rdf:type :"+className+".} ";
```

```
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    ArrayList theList = allios.getArrayLisFromQueryResults(results);
    return theList;
}


 static ArrayList  getMembersOfClassInArrayList(Model theModel, String className)
{
    String theQuery = allios.getMyPrefix()+
                    "SELECT ?a  WHERE { ?a rdf:type :"+className+" }";
        Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    ArrayList theList = allios.getArrayLisFromQueryResults(results);
    return theList;
}




static ArrayList  getAllIndividualsOfSuperClass(Model theModel, String className)
{
    String theQuery = allios.getMyPrefix()+
        "SELECT ?b  WHERE {  ?a rdfs:subClassOf :"+className+".    ?b rdf:type ?a}";
    Query query = QueryFactory.create(theQuery);
        QueryExecution qe = QueryExecutionFactory.create(query, theModel);
        ResultSet results = qe.execSelect();
    ArrayList theList = allios.getArrayLisFromQueryResults(results);
    return theList;
}
```

```
static ObservableList<String> getMembersOfClass(String theFile, String className)
{
   Model theModel = ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM);
   theModel.read(theFile);
   String theQuery = allios.getMyPrefix()+
                  "SELECT ?a  WHERE { ?a rdf:type :"+className+" }";
       Query query = QueryFactory.create(theQuery);
       QueryExecution qe = QueryExecutionFactory.create(query, theModel);
       ResultSet results = qe.execSelect();
   ObservableList<String> theList = allios.getObervableLisFromQueryResults(results);
   return theList;
}

static Exception updateQueryOnTTLFile(String myFile, String queryString) throws Exception
{
   allios.Say("This is updateQueryOnTTLFile");
   allios.Say("This is updateQueryOnTTLFile " + myFile);
   allios.Say("Query :\n " + queryString);
   Exception myEx = null;
   try
   {
      File f = new File(myFile);
      String path = f.getAbsolutePath();
      Model theModel =
       ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM).read(path);

//          Model model = ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM);
      theModel.read(myFile);
      UpdateAction.parseExecute( queryString, theModel );
      try{allios.WriteModelToTurtle(theModel,myFile);}
      catch(Exception exception){myEx = exception;}
   }
```

```
    catch(Exception exception){myEx = exception;}
return     myEx;
}

static ObservableList<String> getObervableLisFromQueryResults(ResultSet results)
{
    try
    {
        QuerySolution querySolution;
            ArrayList<String> list=new ArrayList<>();
            String tmp;
        List<String> vn = results.getResultVars();
        while (results.hasNext())
        {
            querySolution = results.nextSolution();
            Iterator<String> vnames = querySolution.varNames();
                    tmp = querySolution.get(vnames.next()).toString();
            list.add(allios.clippHTTP(tmp));
        }
            return  FXCollections.observableArrayList(list);
    }
    catch( Exception e)
     {
         System.out.println(e.toString());
         return null;
     }
}

static ArrayList<String> getArrayLisFromQueryResults(ResultSet results)
{
    try
    {
        QuerySolution querySolution;
```

```
            ArrayList<String> list=new ArrayList<>();
            String tmp;
        List<String> vn = results.getResultVars();
        while (results.hasNext())
        {
           querySolution = results.nextSolution();
           Iterator<String> vnames = querySolution.varNames();
                   tmp = querySolution.get(vnames.next()).toString();
           list.add(allios.clippHTTP(tmp));
        }
           return  list;
    }
    catch( Exception e)
     {
        System.out.println(e.toString());
        return null;
     }
}


static String clippHTTP(String uri)//P
{   return uri.substring((uri.indexOf("#"))+1);}


static Exception renameAll(String theFile, String oldName, String newName)
{
    Exception myEx = null;
    try
    {
        String queryString = allios.getMyPrefix()+
                        "delete  \n" +
                        " { :"+ oldName +" ?a ?b} \n" +
                        "insert  \n" +
```

```
                    " { :" +newName+" ?a ?b}\n"  +
                    "where { :"+ oldName +" ?a ?b};\n";
        allios.updateQueryOnTTLFile(theFile, queryString);

        }
        catch(Exception e){myEx=e;}
        return myEx;
    }


}
```

Αρχείο: “splash2.fxml”

```
<?xml version="1.0" encoding="UTF-8"?>


<?import javafx.scene.control.Label?>
<?import javafx.scene.control.ListView?>
<?import javafx.scene.control.Menu?>
<?import javafx.scene.control.MenuBar?>
<?import javafx.scene.control.MenuItem?>
<?import javafx.scene.control.SeparatorMenuItem?>
<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.VBox?>
<?import javafx.scene.text.Font?>


<VBox fx:id="VBsplash" prefHeight="531.0" prefWidth="709.0" xmlns="http://javafx.com/javafx/8.0.171" xmlns:fx="http://javafx.com/fxml/1" fx:controller="beta004.Splash2Control">
  <children>
    <MenuBar fx:id="splash_menu_bar" VBox.vgrow="NEVER">
      <menus>
        <Menu fx:id="menu_file" mnemonicParsing="false" text="Αρχείο Δεδομένων">
          <items>
            <MenuItem fx:id="ExportToTTF" mnemonicParsing="false" onAction="#runExportToTTF" text="Εξαγωγή Οντολογίας σε Turtle Αρχείο" />
            <MenuItem fx:id="ImportFromFile" mnemonicParsing="false" onAction="#runImportFromFile" text="Εισαγωγή Οντολογίας απο Αρχείο" />
            <SeparatorMenuItem mnemonicParsing="false" />
            <MenuItem fx:id="StartOver" mnemonicParsing="false" onAction="#runStartOver" text="Διαγραφή Όλων (start over)" />
            <MenuItem fx:id="StartOverWithSampleData" mnemonicParsing="false" onAction="#runStartOverWithSampleData" text="Διαγραφή Όλων και Φόρτωση Δεδομένων Ελέγχου" />
            <MenuItem fx:id="MNexitApp" mnemonicParsing="false" onAction="#doExitApp" text="Εξοδος" />
```

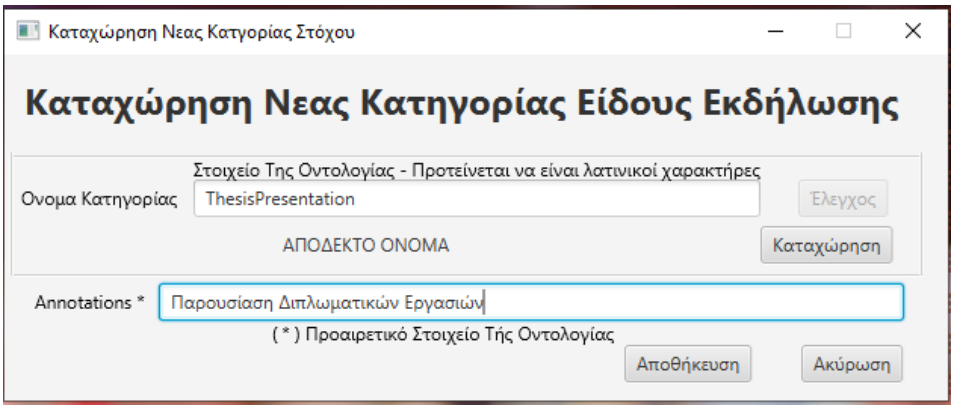
```
        </items>
      </Menu>
      <Menu fx:id="menu_data" mnemonicParsing="false" text="Διαχείρηση Δεδομένων">
        <items>
          <MenuItem fx:id="addNewHappeningNoEvent" mnemonicParsing="false"
onAction="#doAddNewEntityNoHappening" text="Νεα Οντότητα Χωρίς Συμβάν" />
          <MenuItem fx:id="MN_editEntity" mnemonicParsing="false" onAction="#runEditEntity"
text="Επεξεργασια Οντότητας" />
          <MenuItem fx:id="MInewEvent" mnemonicParsing="false" onAction="#loadNewEvent"
text="Νεα εκδήλωση" />
          <MenuItem fx:id="MNnewHappNoEvent" mnemonicParsing="false"
onAction="#runAddNewHappeningNoEvent" text="Νεο Συμβάν Χωρίς Εκδήλωση" />
          <MenuItem fx:id="MNnewProductIndividual" mnemonicParsing="false"
onAction="#newProductIndividual" text="Νεο Προιόν Εκδήλωσης" />
          <SeparatorMenuItem mnemonicParsing="false" />
          <MenuItem fx:id="MNnewPlace" mnemonicParsing="false" onAction="#openAddNewPlace"
text="Νεος Χώρος Εκδήλωσης" />
          <SeparatorMenuItem mnemonicParsing="false" />
        </items>
      </Menu>
      <Menu fx:id="menu_onto" mnemonicParsing="false" text="Οντολογική Διαχείρηση">
        <items>
          <MenuItem fx:id="MNnewProductClass" mnemonicParsing="false"
onAction="#runNewProductClass" text="Νεα Κατηγορία Παραγώγου Προιόντος Εκδήλωσης" />
          <MenuItem fx:id="MNnewScopeClass" mnemonicParsing="false"
onAction="#runNewScopeClass" text="Νεα Κατηγορία Σκοπού" />
          <SeparatorMenuItem mnemonicParsing="false" />
          <MenuItem fx:id="MNnewParticipantClass" mnemonicParsing="false"
onAction="#runNewParticipantClass" text="Νεα κατηγορία Συμμετέχοντος" />
          <MenuItem fx:id="MNnewParticipantIndividual" mnemonicParsing="false"
onAction="#addNewParticipantIndividual" text="      Νεος Ρόλος Σε Κατηγορία" />
          <SeparatorMenuItem mnemonicParsing="false" />
          <MenuItem fx:id="MNrenameClass" mnemonicParsing="false"
onAction="#runRenameClass" text="Μαζική Μετονομασία Κατηγορίας" />
          <MenuItem fx:id="MNrnameIndividuals" mnemonicParsing="false"
onAction="#runRenameIndividuals" text="Μαζική Μετονομασία ..." />
```

```
            <SeparatorMenuItem mnemonicParsing="false" />
            <MenuItem fx:id="MNqueries" mnemonicParsing="false" onAction="#showQueries" text="Ελεύθερα Ερωτήματα" />
            <MenuItem fx:id="MNpreMadeQueries" mnemonicParsing="false" onAction="#showPreMadeQueries" text="Προκατασκευασμένα Ερωτήματα" />
          </items>
        </Menu>
        <Menu fx:id="MNabout" mnemonicParsing="false" onAction="#handleMenuFC" text="Πληροφορίες Εκδοσης" />
      </menus>
    </MenuBar>
    <ListView fx:id="LVH1" />
    <AnchorPane fx:id="APmain" maxHeight="-1.0" maxWidth="-1.0" prefHeight="394.0" prefWidth="649.0" VBox.vgrow="ALWAYS">
      <children>
        <ImageView fx:id="imgBack" fitHeight="523.0" fitWidth="716.0" layoutX="-8.0" layoutY="5.0" pickOnBounds="true">
          <image>
            <Image url="@../IMG/event.jpg" />
          </image>
        </ImageView>
        <Label layoutX="3.0" layoutY="69.0" text="Με Χρήση Οντολογιών" textFill="#f8f8f8">
          <font>
            <Font name="System Bold" size="36.0" />
          </font>
        </Label>
        <Label fx:id="lblM01_shadow" layoutX="10.0" layoutY="124.0" text="ΔΙΑΘΕΣΙΜΕΣ ΕΚΔΗΛΩΣΕΙΣ :" textFill="#000d00" wrapText="true">
          <font>
            <Font name="System Bold" size="24.0" />
          </font>
        </Label>
        <Label fx:id="lblM01" layoutY="114.0" text="ΔΙΑΘΕΣΙΜΕΣ ΕΚΔΗΛΩΣΕΙΣ :" textFill="#fefffe" wrapText="true">
          <font>
```

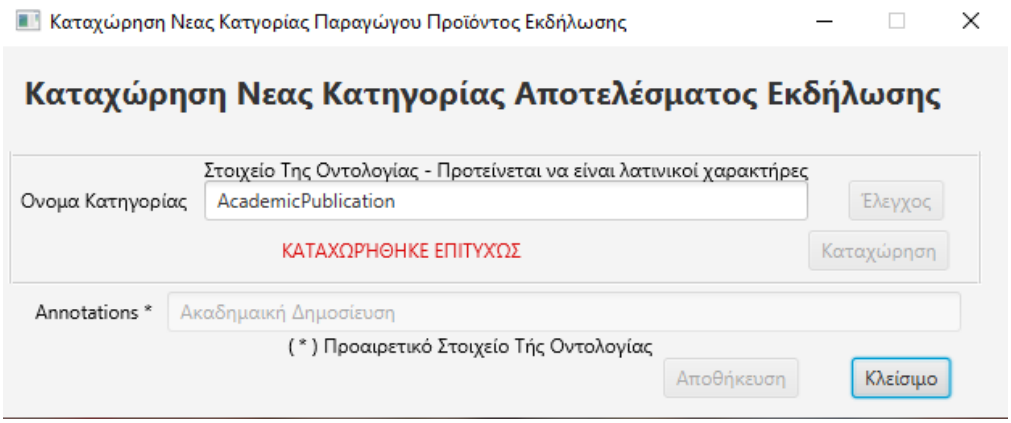
```
            <Font name="System Bold" size="24.0" />
          </font>
        </Label>
        <Label fx:id="lblM2_shadow" layoutX="30.0" layoutY="189.0" text="ΔΙΑΘΕΣΙΜΑ ΣΥΜΒΑΝΤΑ :" textFill="#0a0a0a" wrapText="true">
          <font>
            <Font name="Times New Roman Bold Italic" size="24.0" />
          </font>
        </Label>
        <Label fx:id="lblM2" layoutX="20.0" layoutY="179.0" text="ΔΙΑΘΕΣΙΜΑ ΣΥΜΒΑΝΤΑ:" textFill="#e4c7b6" wrapText="true">
          <font>
            <Font name="Times New Roman Bold Italic" size="24.0" />
          </font>
        </Label>
        <Label fx:id="lblM3_shadow" layoutX="30.0" layoutY="250.0" text="ΔΙΑΘΕΣΙΜΕΣ ΟΝΤΟΤΗΤΕΣ :" textFill="#171717" wrapText="true">
          <font>
            <Font name="Arial Black" size="18.0" />
          </font>
        </Label>
        <Label fx:id="lblM3" layoutX="20.0" layoutY="240.0" text="ΔΙΑΘΕΣΙΜΕΣ ΟΝΤΟΤΗΤΕΣ :" textFill="#e84d2a" wrapText="true">
          <font>
            <Font name="Arial Black" size="18.0" />
          </font>
        </Label>
        <Label layoutX="3.0" layoutY="-6.0" text="Διαχείρηση Εκδηλώσεων " textFill="#070607">
          <font>
            <Font name="System Bold" size="64.0" />
          </font>
        </Label>
        <Label layoutY="-8.0" text="Διαχείρηση Εκδηλώσεων " textFill="#1b0bd3">
          <font>
```

```
            <Font name="System Bold" size="64.0" />
          </font>
        </Label>
        <Label layoutX="1.0" layoutY="68.0" text="Με Χρήση Οντολογιών" textFill="#bcb9e4">
          <font>
            <Font name="System Bold" size="36.0" />
          </font>
        </Label>
      </children>
    </AnchorPane>
  </children>
</VBox>
```

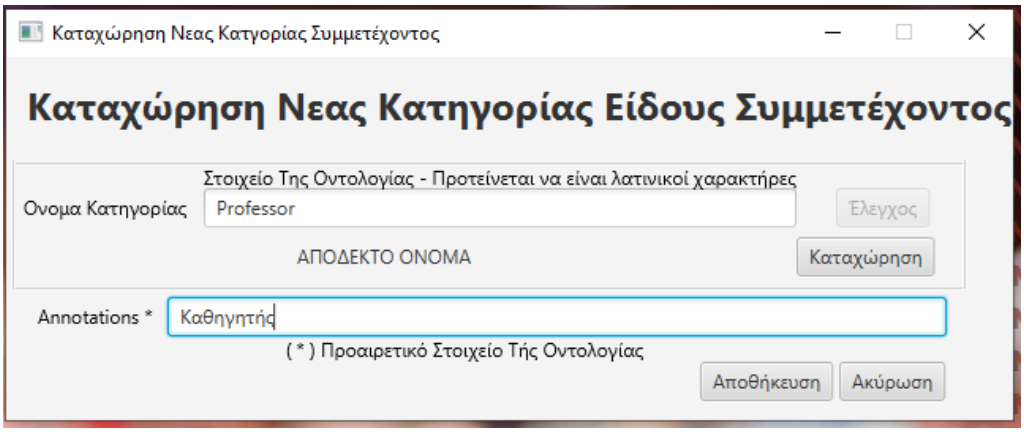
ΠΑΡΑΡΤΗΜΑ D Στιγμιότυπα οθόνης



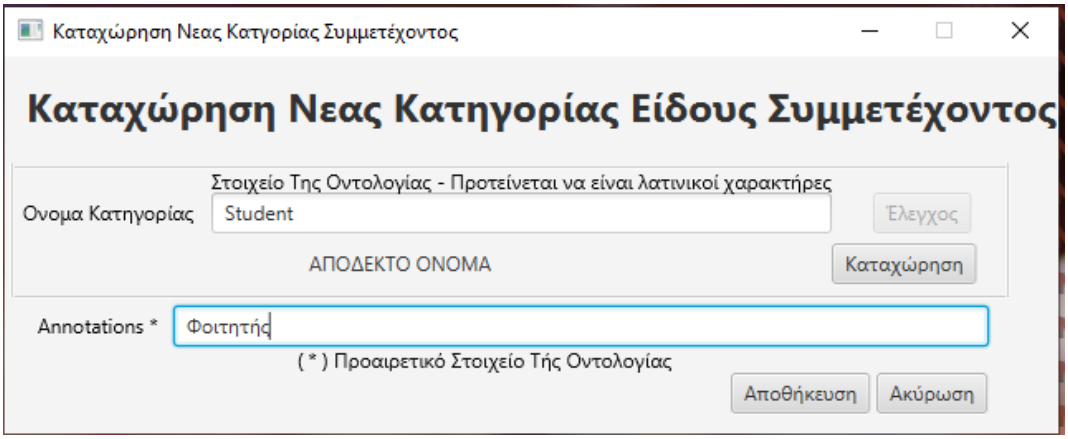

```
base:ThesisPresentation
        a               owl:Class ;
        rdfs:comment    "Παρουσίαση Διπλωματικών Εργασιών" ;
        rdfs:subClassOf base:Scope .
```



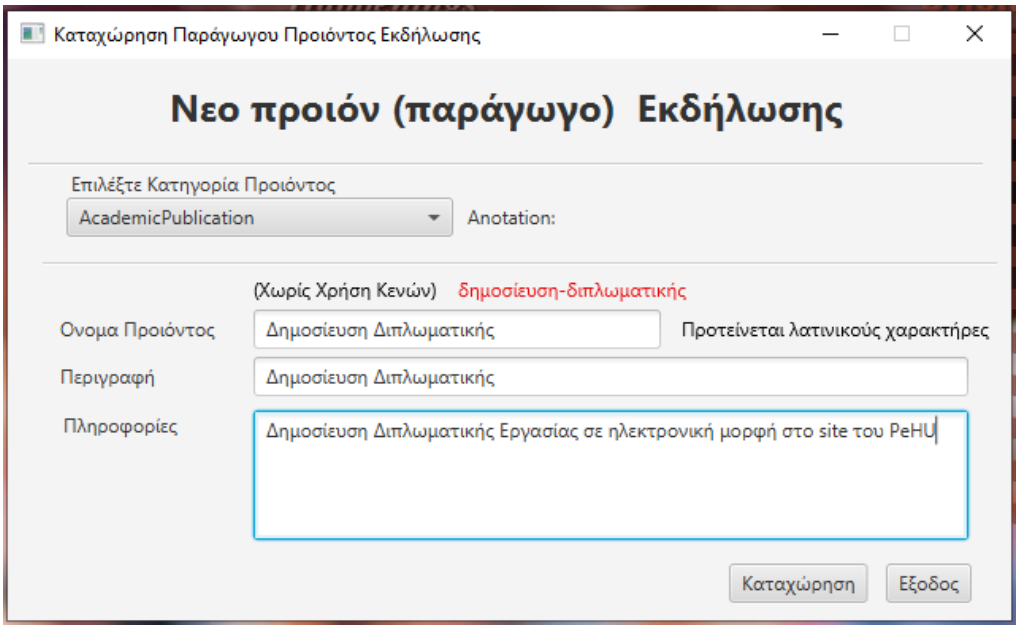

```
base:AcademicPublication
        a               owl:Class ;
        rdfs:comment    "Ακαδημαική Δημοσίευση" ;
        rdfs:subClassOf base:Product .
```



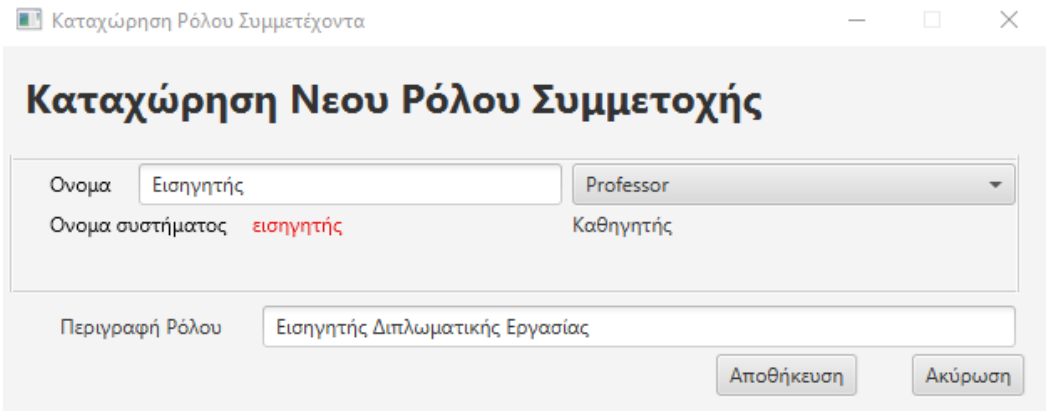

```
base:Professor  a           owl:Class ;
        rdfs:comment        "Καθηγητής" ;
        rdfs:subClassOf     base:Participant .
```



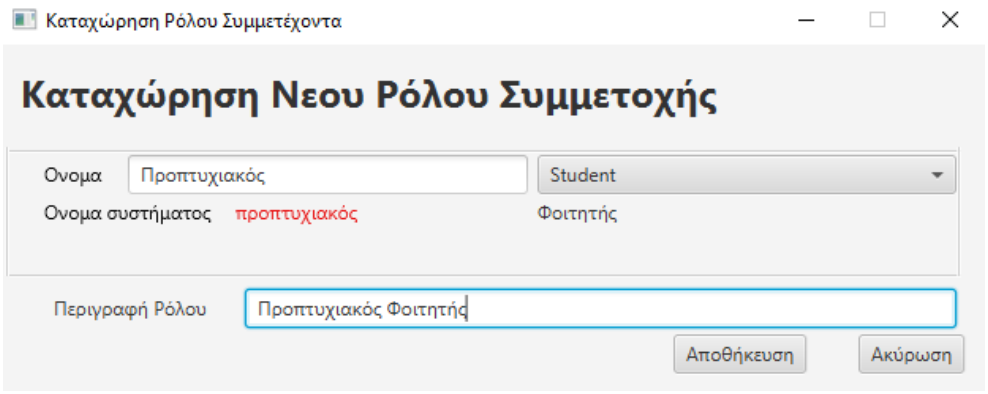

```
base:Student  a           owl:Class ;
        rdfs:comment      "Φοιτητής" ;
        rdfs:subClassOf   base:Participant .
```



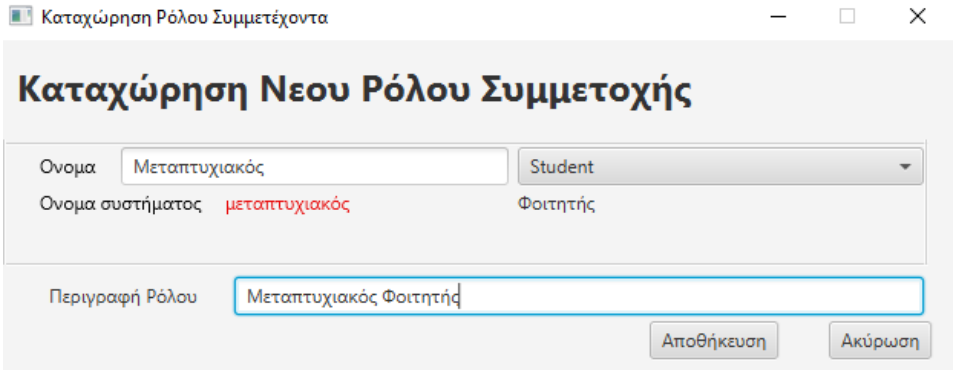

```
base:δημοσίευση-διπλωματικής
        a                       owl:NamedIndividual , base:AcademicPublication ;
        base:scopeDescription   "Δημοσίευση Διπλωματικής" ;
        base:scopeInfo          "Δημοσίευση Διπλωματικής Εργασίας σε ηλεκτρονική μορφή στο site του PeHU" .
```



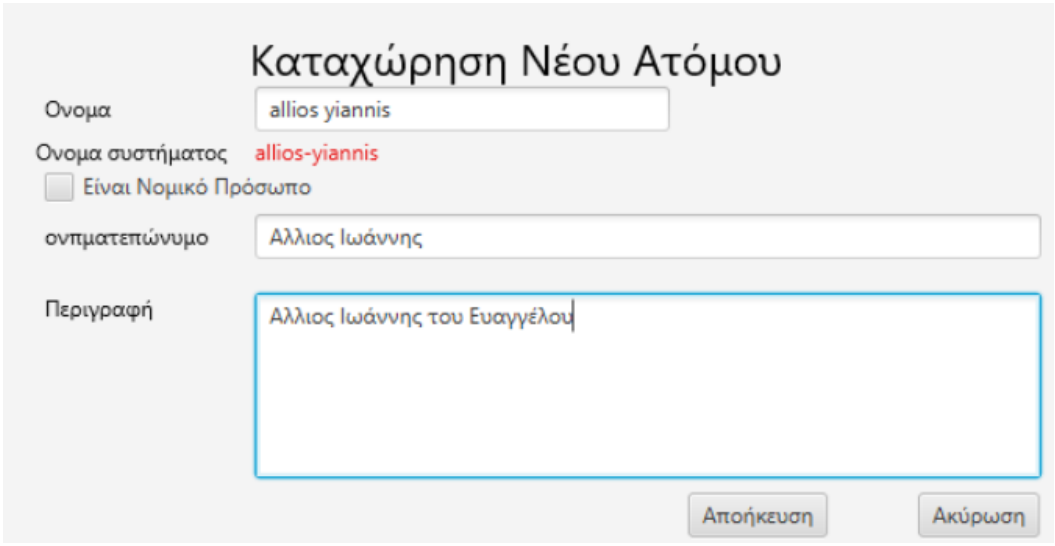

```
base:εισηγητής  a                   owl:NamedIndividual , base:Professor ;
        base:participationInfo  "Εισηγητής Διπλωματικής Εργασίας" .
```



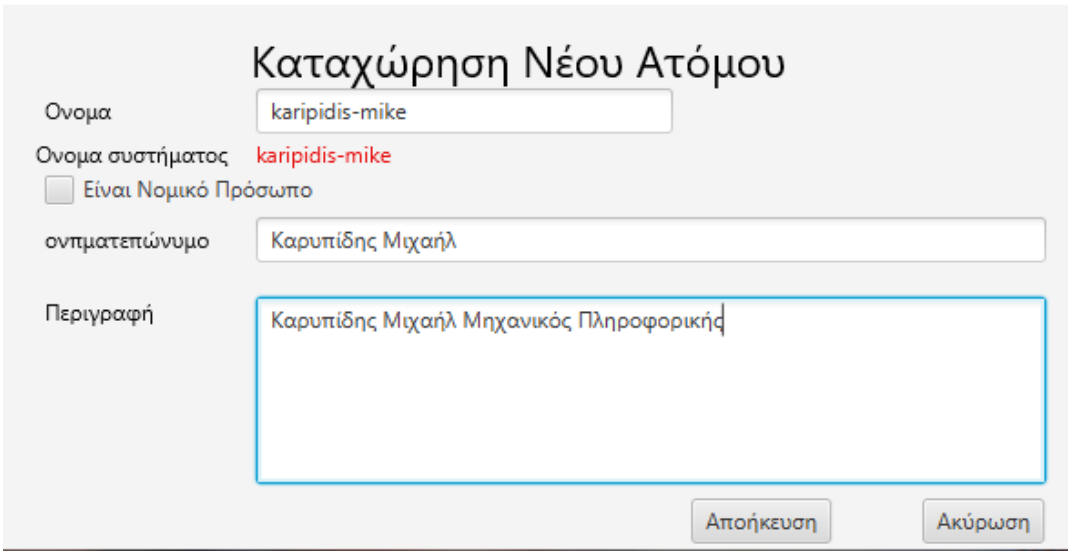

```
base:προπτυχιακός  a                owl:NamedIndividual , base:Student ;
       base:participationInfo  "Προπτυχιακός Φοιτητής" .
```



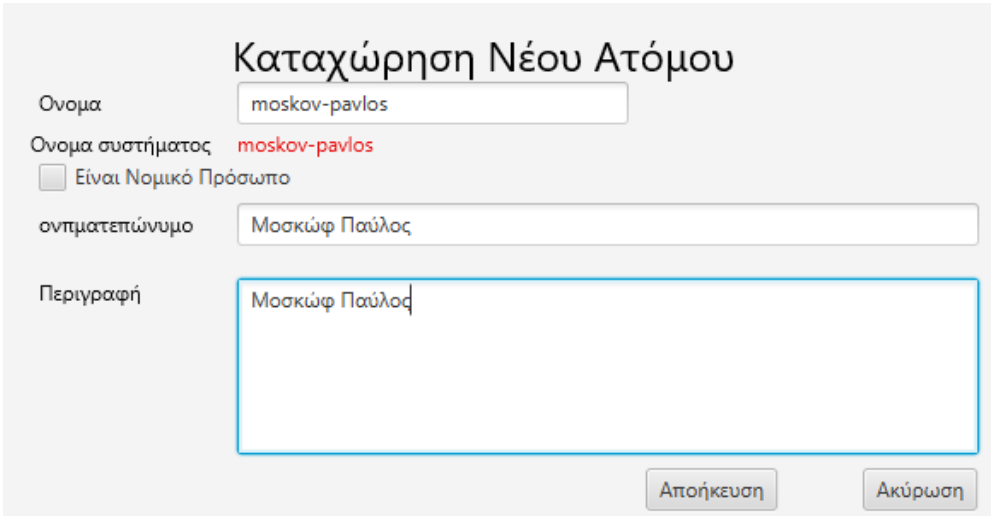

```
base:μεταπτυχιακός  a                owl:NamedIndividual , base:Student ;
       base:participationInfo  "Μεταπτυχιακός Φοιτητής" .
```



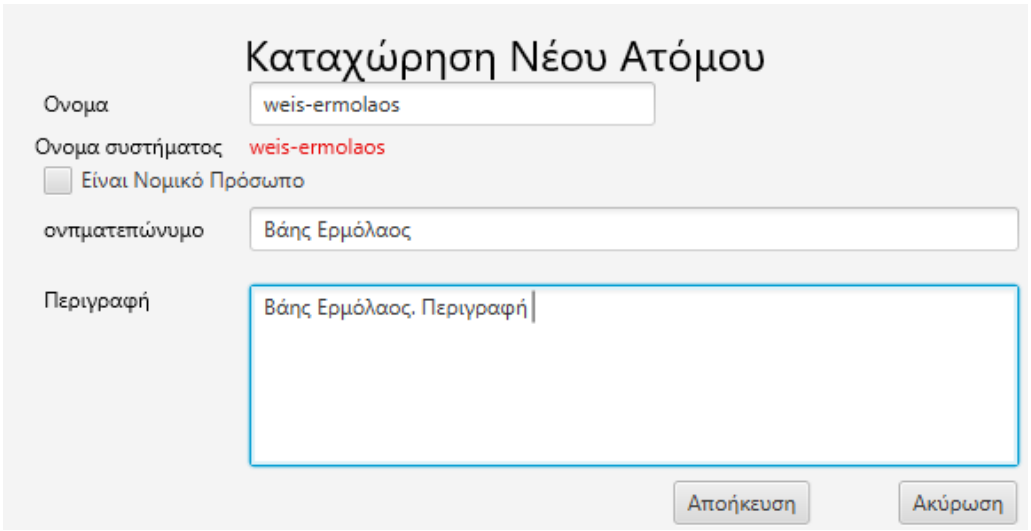

```
base:allios-yiannis  a                  owl:NamedIndividual , base:Physical ;
       base:EntityDescription   "Αλλιος Ιωάννης του Ευαγγέλου" ;
       base:EntityPhisicalName  "Αλλιος Ιωάννης" .
```



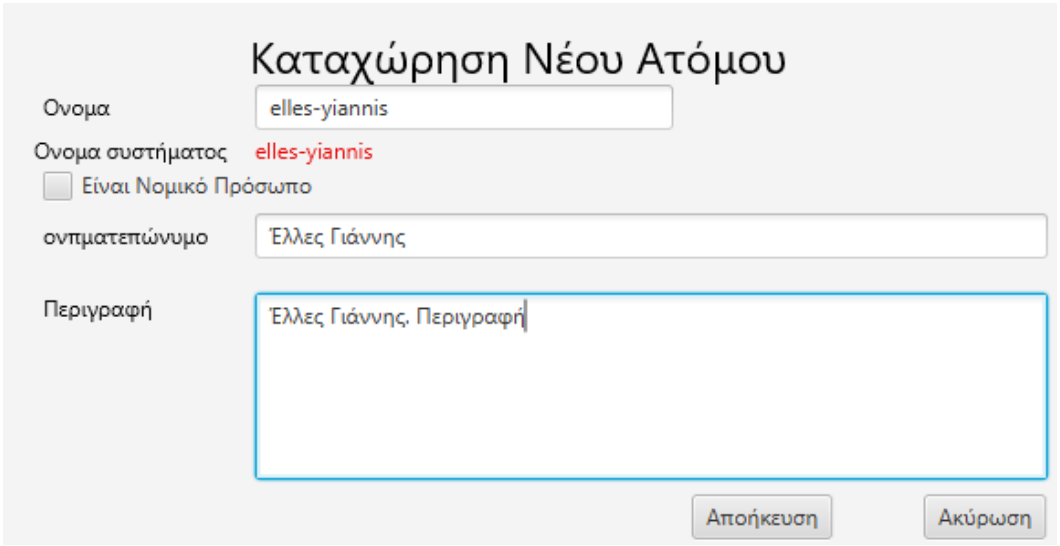

```
base:karipidis-mike  a                  owl:NamedIndividual , base:Physical ;
       base:EntityDescription   "Καρυπίδης Μιχαήλ Μηχανικός Πληροφορικής" ;
       base:EntityPhisicalName  "Καρυπίδης Μιχαήλ" .
```



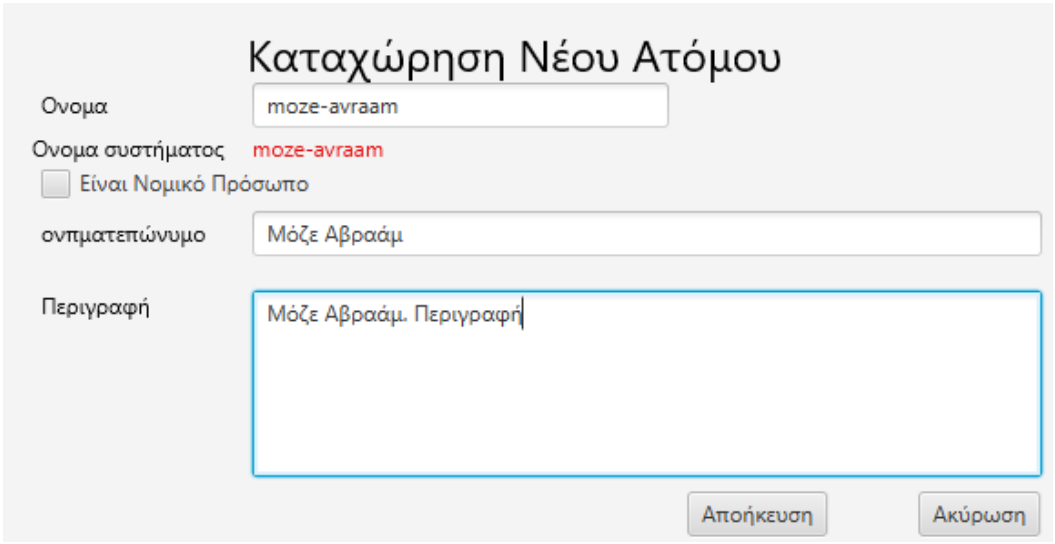

```
base:moskov-pavlos  a                   owl:NamedIndividual , base:Physical ;
        base:EntityDescription          "Μοσκώφ Παύλος" ;
        base:EntityPhisicalName         "Μοσκώφ Παύλος" .
```



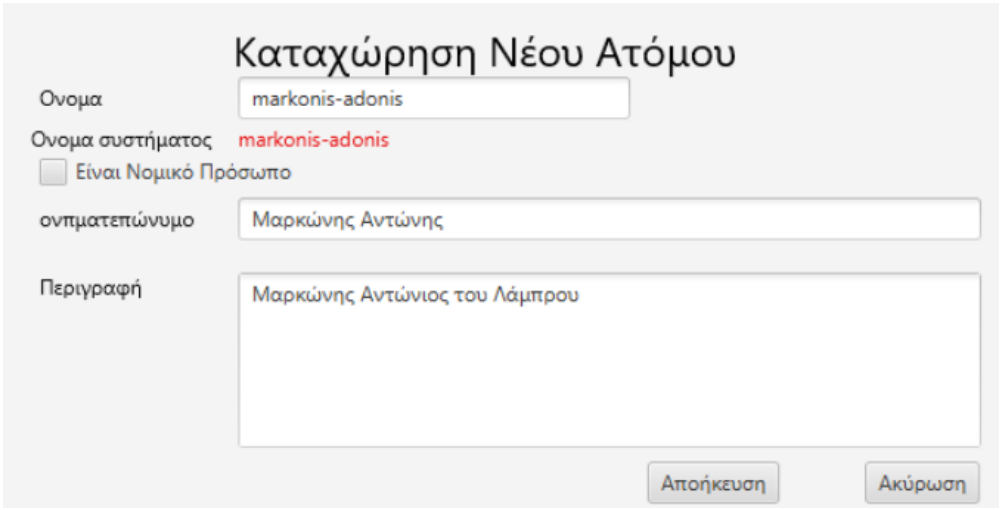

```
base:weis-ermolaos  a                   owl:NamedIndividual , base:Physical ;
        base:EntityDescription          "Βάης Ερμόλαος. Περιγραφή " ;
        base:EntityPhisicalName         "Βάης Ερμόλαος" .
```



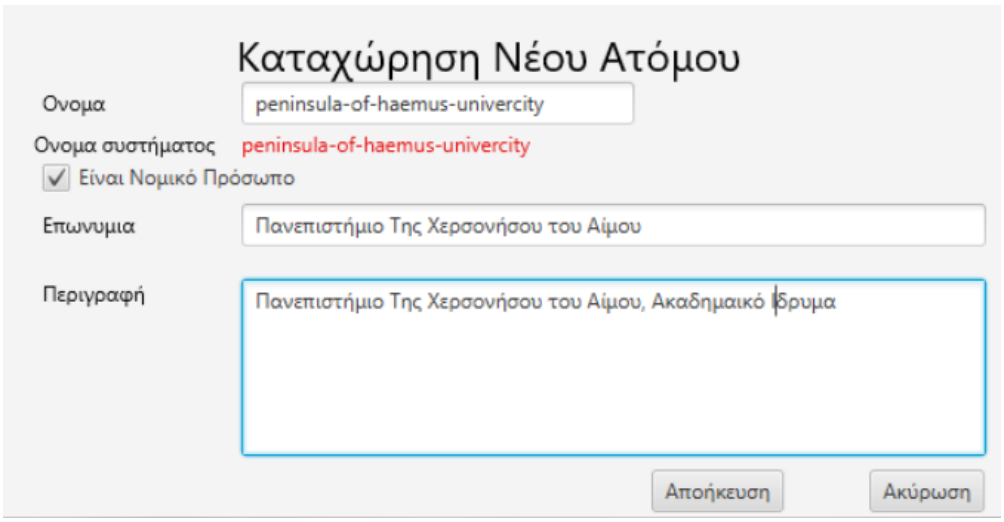

```
base:elles-yiannis  a                  owl:NamedIndividual , base:Physical ;
        base:EntityDescription         "Έλλες Γιάννης. Περιγραφή" ;
        base:EntityPhisicalName        "Έλλες Γιάννης" .
```



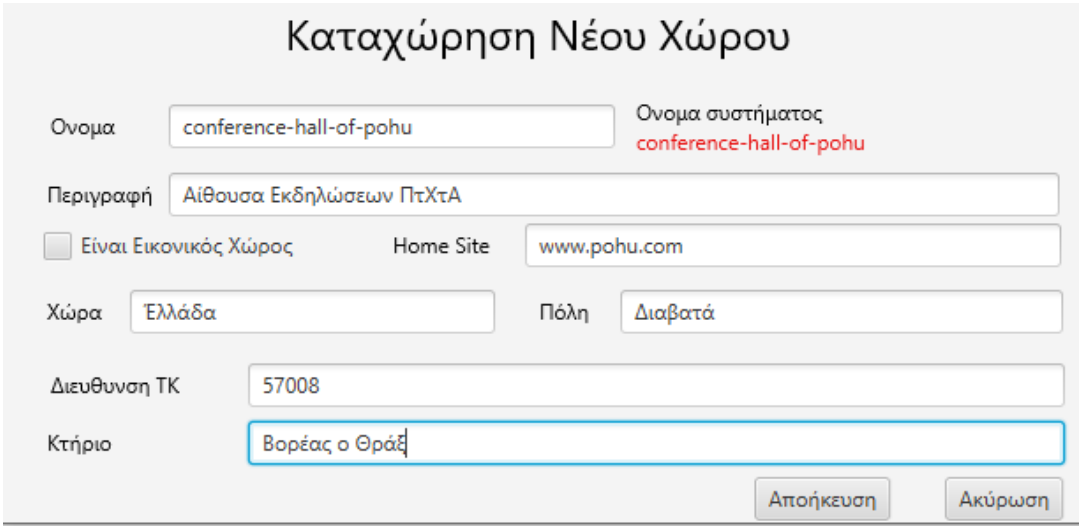

```
base:moze-avraam  a                    owl:NamedIndividual , base:Physical ;
        base:EntityDescription         "Μόζε Αβραάμ. Περιγραφή" ;
        base:EntityPhisicalName        "Μόζε Αβραάμ" .
```



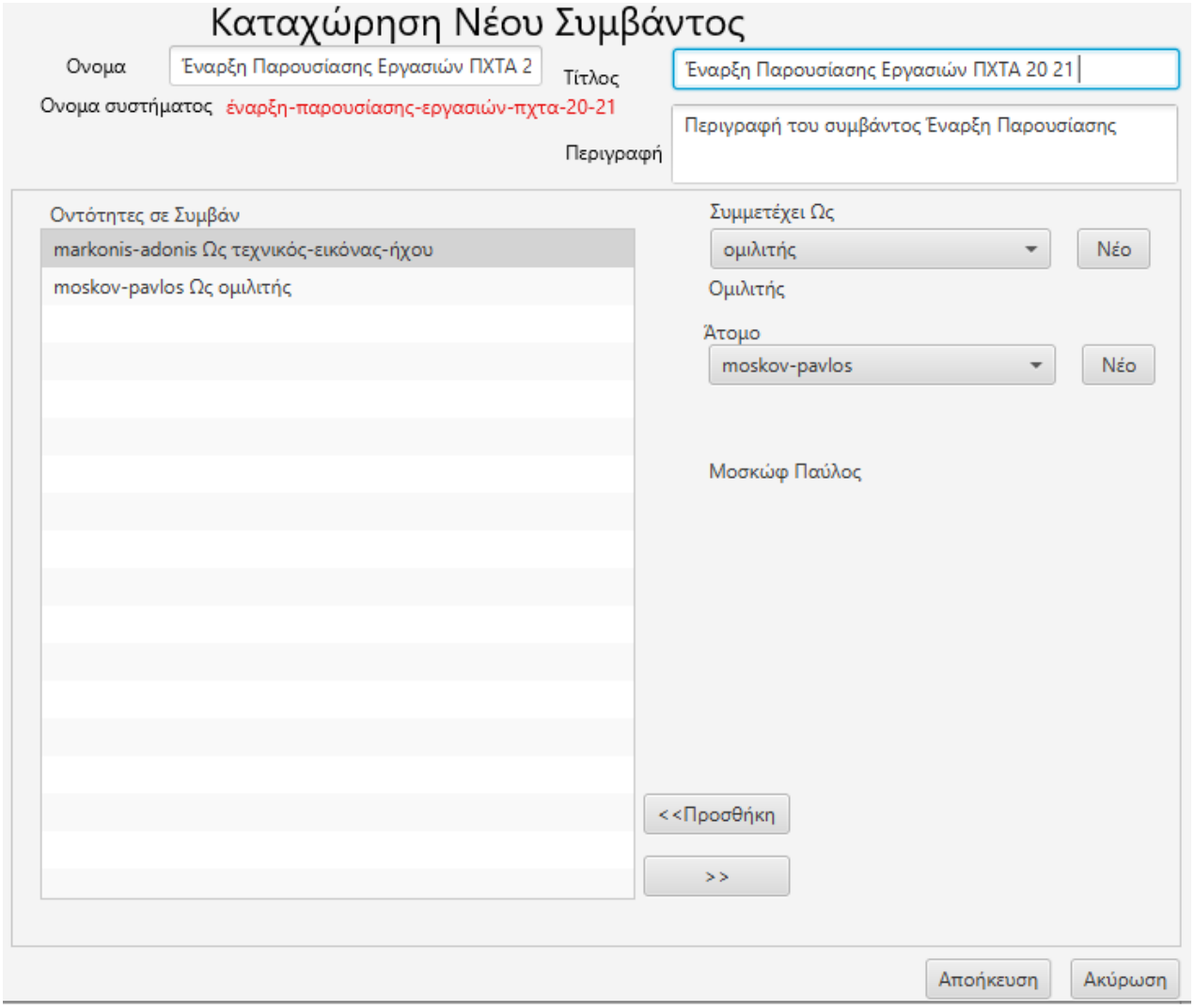

```
base:markonis-adonis  a                    owl:NamedIndividual , base:Physical ;
        base:EntityDescription   "Μαρκώνης Αντώνιος του Λάμπρου" ;
        base:EntityPhisicalName  "Μαρκώνης Αντώνης" .
```



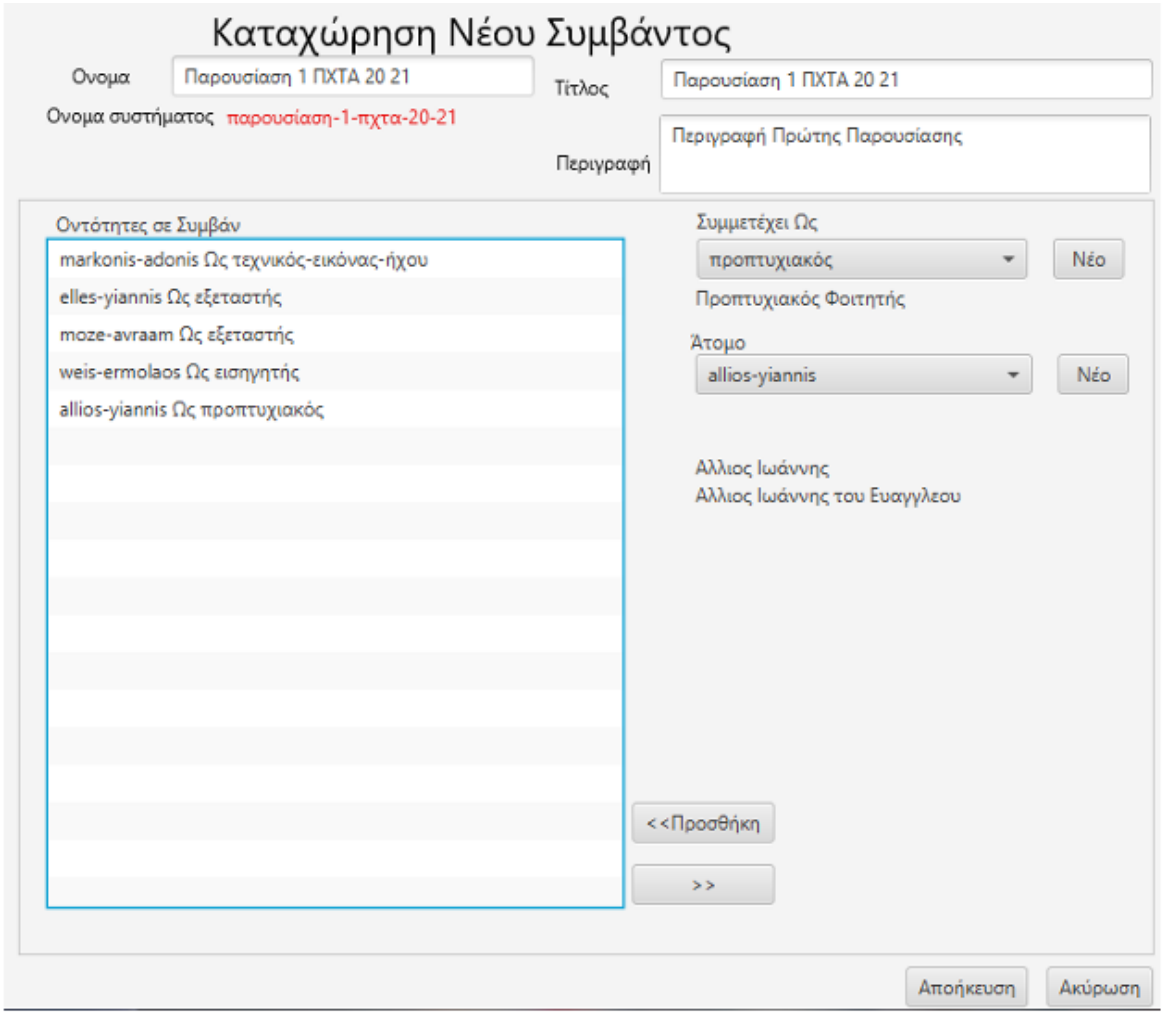

```
base:peninsula-of-haemus-univercity
        a                        base:Physical , owl:NamedIndividual ;
        base:EntityDescription   "Πανεπιστήμιο της Χερσονήσου του Αίμου, Ακαδημαικό Ίδρυμα" ;
        base:EntityPhisicalName  "Πανεπιστήμιο της Χερσονήσου του Αίμου" .
```



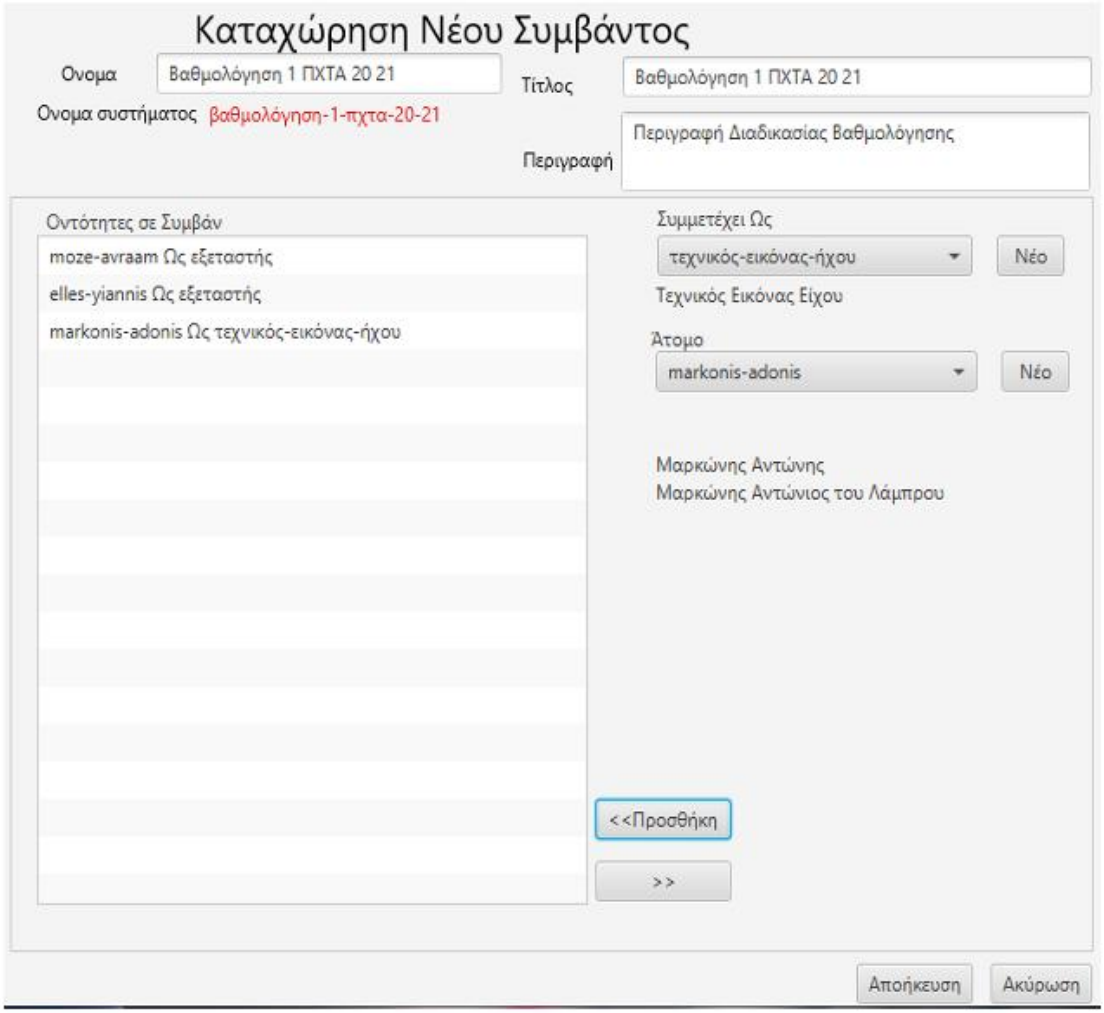

```
base:conference-hall-of-pohu
        a                       owl:NamedIndividual , base:Actual ;
        base:PlaceAddress       "57008" ;
        base:PlaceBuilding      "Βορέας ο Θράξ" ;
        base:PlaceCity          "Διαβατά" ;
        base:PlaceCountry       "Ἑλλάδα" ;
        base:PlaceDiscription   "Αίθουσα Εκδηλώσεων ΠτΧτΑ" ;
        base:PlaceWebAddress    "www.pohu.com" .
```



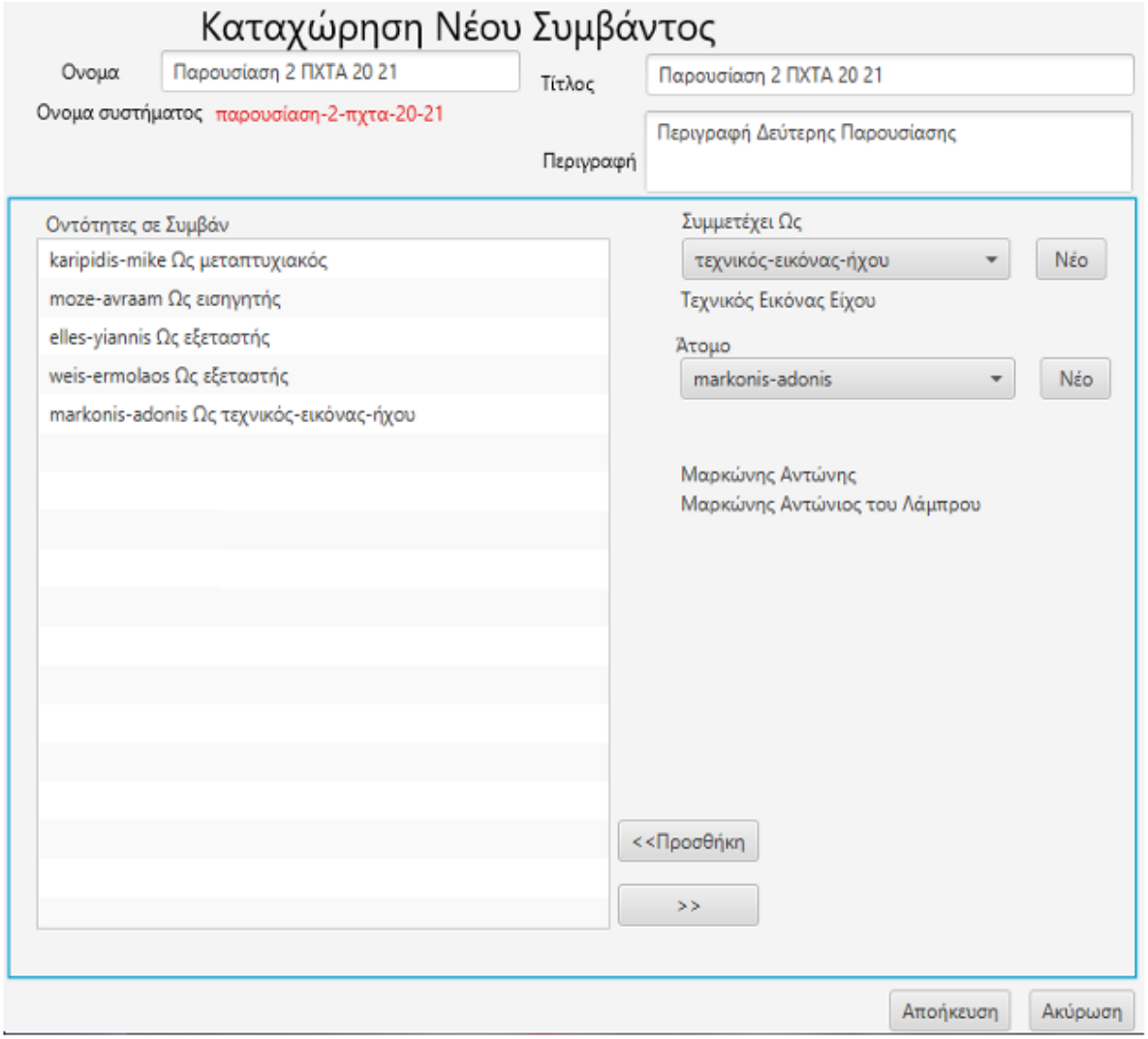

```
base:έναρξη-παρουσίασης-εργασιών-πχτα-20-21
        a                       owl:NamedIndividual , base:Happening ;
        base:EntityAsParticipant  "moskov-pavlos Ως ομιλιτής" , "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ;
        base:HappDescription      "Περιγραφή του συμβάντος Έναρξη Παρουσίασης" ;
        base:HappInfo             "Έναρξη Παρουσίασης Εργασιών ΠΧΤΑ 20 21 " ;
        base:hasAsFactor          base:moskov-pavlos , base:markonis-adonis .
```



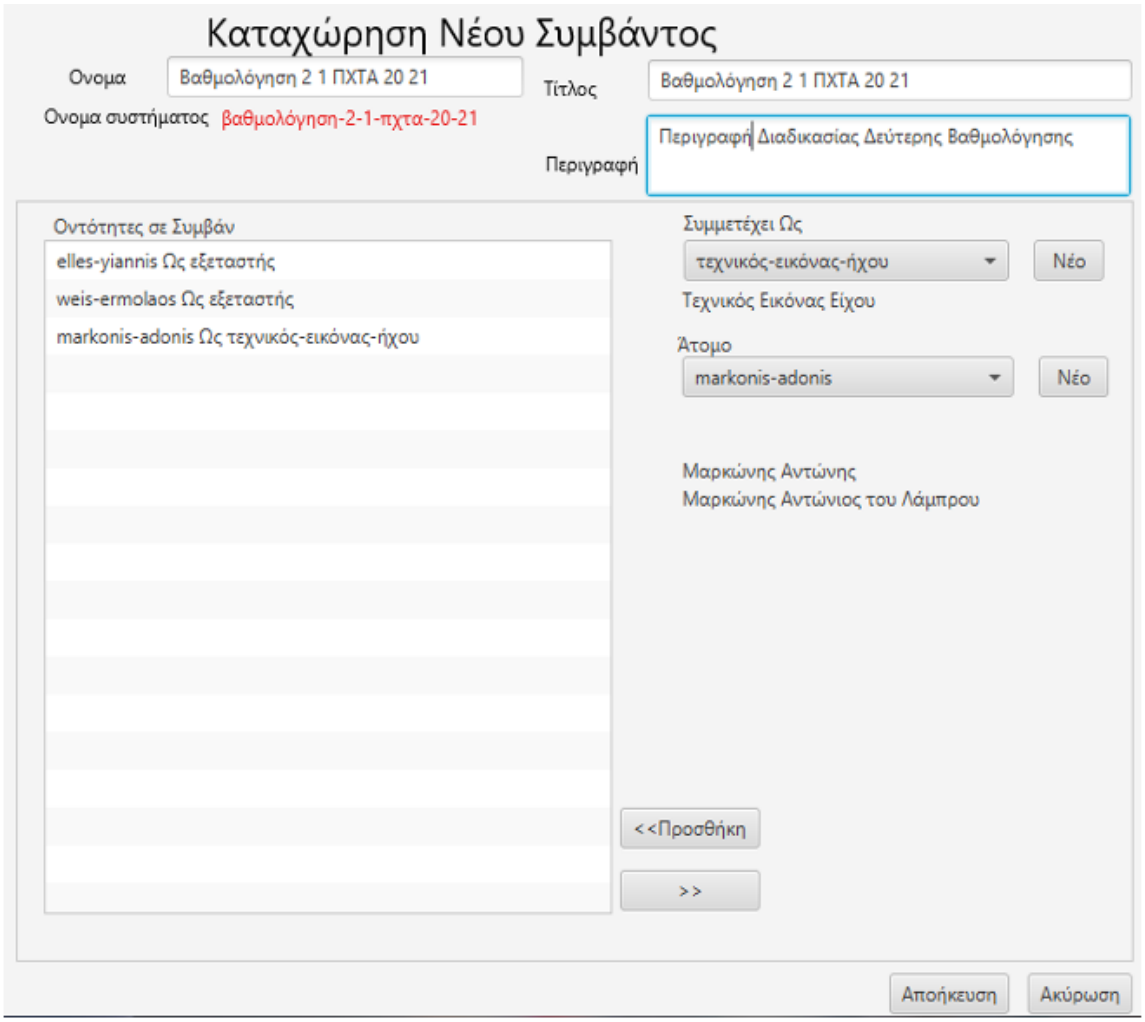

```
base:παρουσίαση-1-πχτα-20-21
        a                           owl:NamedIndividual ,
                                    base:Happening ;
        base:EntityAsParticipant    "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ,
                                    "weis-ermolaos Ως εισηγητής" ,
                                    "elles-yiannis Ως εξεταστής" ,
                                    "allios-yiannis Ως προπτυχιακός" ,
                                    "moze-avraam Ως εξεταστής" ;
        base:HappDescription        "Περιγραφή Πρώτης Παρουσίασης " ;
        base:HappInfo               "Παρουσίαση 1 ΠΧΤΑ 20 21" ;
        base:hasAsFactor            base:moze-avraam ,
                                    base:allios-yiannis ,
                                    base:elles-yiannis ,
                                    base:weis-ermolaos ,
                                    base:markonis-adonis .
```



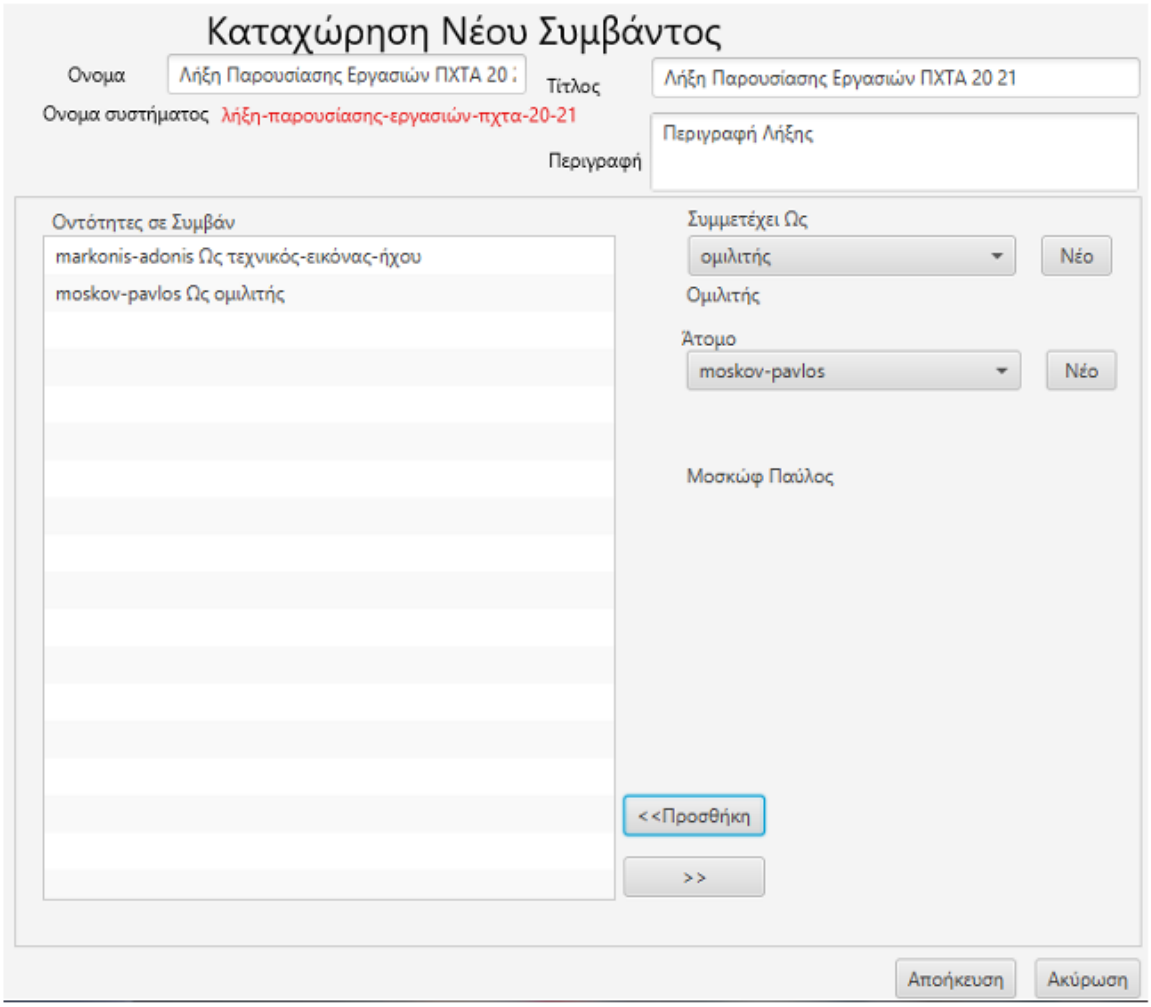

```
base:βαθμολόγηση-1-πχτα-20-21
        a                               base:Happening ,
                                        owl:NamedIndividual ;
        base:EntityAsParticipant        "moze-avraam Ως εξεταστής" ,
                                        "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ,
                                        "elles-yiannis Ως εξεταστής" ;
        base:HappDescription            "Περιγραφή Διαδικασίας Βαθμολόγησης" ;
        base:HappInfo                   "Βαθμολόγηση 1 ΠΧΤΑ 20 21 " ;
        base:hasAsFactor                base:elles-yiannis ,
                                        base:moze-avraam ,
                                        base:markonis-adonis .
```



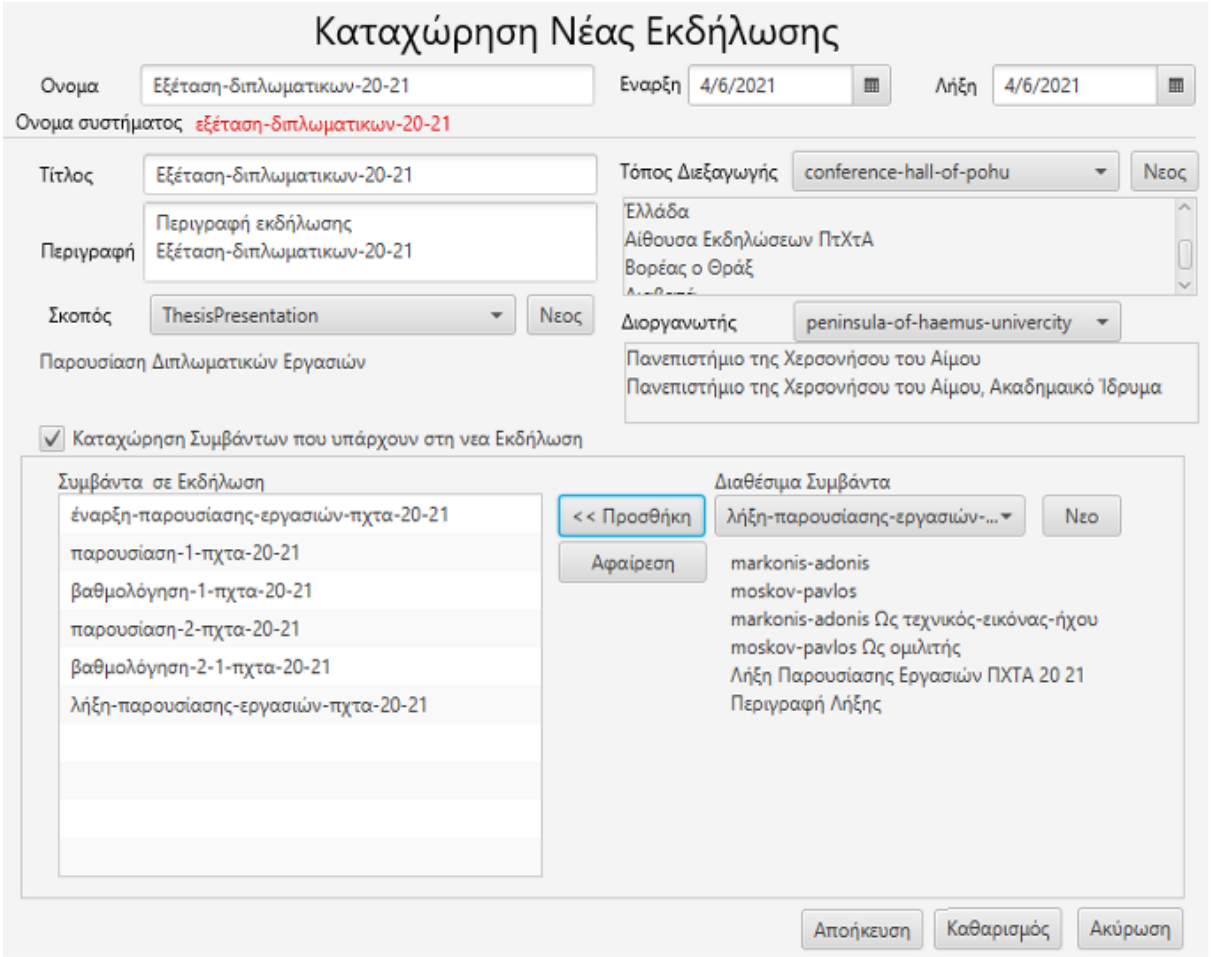

```
base:παρουσίαση-2-πχτα-20-21
        a                               base:Happening ,
                                        owl:NamedIndividual ;
        base:EntityAsParticipant        "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ,
                                        "karipidis-mike Ως μεταπτυχιακός" ,
                                        "weis-ermolaos Ως εξεταστής" ,
                                        "moze-avraam Ως εισηγητής" ,
                                        "elles-yiannis Ως εξεταστής" ;
        base:HappDescription            "Περιγραφή Δεύτερης Παρουσίασης" ;
        base:HappInfo                   "Παρουσίαση 2 ΠΧΤΑ 20 21 " ;
        base:hasAsFactor                base:elles-yiannis ,
                                        base:moze-avraam ,
                                        base:karipidis-mike ,
                                        base:weis-ermolaos ,
                                        base:markonis-adonis .
```




```
base:βαθμολόγηση-2-1-πχτα-20-21
        a                           base:Happening ,
                                    owl:NamedIndividual ;
        base:EntityAsParticipant    "elles-yiannis Ως εξεταστής" ,
                                    "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ,
                                    "weis-ermolaos Ως εξεταστής" ;
        base:HappDescription        "Περιγραφή Διαδικασίας Δεύτερης Βαθμολόγησης" ;
        base:HappInfo               "Βαθμολόγηση 2 1 ΠΧΤΑ 20 21 " ;
        base:hasAsFactor            base:markonis-adonis ,
                                    base:elles-yiannis ,
                                    base:weis-ermolaos .
```




```
base:λήξη-παρουσίασης-εργασιών-πχτα-20-21
        a                               owl:NamedIndividual ,
                                        base:Happening ;
        base:EntityAsParticipant        "moskov-pavlos Ως ομιλιτής" ,
                                        "markonis-adonis Ως τεχνικός-εικόνας-ήχου" ;
        base:HappDescription            "Περιγραφή Λήξης" ;
        base:HappInfo                   "Λήξη Παρουσίασης Εργασιών ΠΧΤΑ 20 21 " ;
        base:hasAsFactor                base:moskov-pavlos ,
                                        base:markonis-adonis .
```




```
base:εξέταση-διπλωματικων-20-21
        a                       owl:NamedIndividual ,
                                base:ThesisPresentation ,
                                base:Event ;
        base:EventBeginDate     "2021-06-04" ;
        base:EventDescription   "Περιγραφή εκδήλωσης  Εξέταση-διπλωματικων-20-21" ;
        base:EventEndDate       "2021-06-04" ;
        base:EventTitle         "Εξέταση-διπλωματικων-20-21" ;
        base:hasHappening       base:έναρξη-παρουσίασης-εργασιών-πχτα-20-21 ,
                                base:παρουσίαση-1-πχτα-20-21 ,
                                base:παρουσίαση-2-πχτα-20-21 ,
                                base:λήξη-παρουσίασης-εργασιών-πχτα-20-21 ,
                                base:βαθμολόγηση-1-πχτα-20-21 ,
                                base:βαθμολόγηση-2-1-πχτα-20-21 ;
        base:hasHost            base:peninsula-of-haemus-univercity ;
        base:hasPlace           base:conference-hall-of-pohu .
```