# ΣΧΕΔΙΑΣΜΟΣ & ΥΛΟΠΟΙΗΣΗ ΤΟΥ GRAMOFONE: ΕΝΑ ΓΡΑΦΙΚΟ ΕΡΓΑΛΕΙΟ ΕΠΕΞΕΡΓΑΣΙΑΣ ΟΝΤΟΛΟΓΙΩΝ ΒΑΣΙΣΜΕΝΟ ΣΤΟ ΠΡΟΤΥΠΟ MOF

**Γιαννόπουλος Νικόλαος**

**Πολυτεχνείο Κρήτης**

**Τμήμα Ηλεκτρονικών Μηχανικών & Μηχανικών Ηλεκτρονικών Υπολογιστών**

Μία εργασία που παρουσιάστηκε στο Πολυτεχνείο Κρήτης για την εκπλήρωση των απαιτήσεων απόκτησης Διπλώματος στο τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών

Χανιά, 2005

*Αφιέρωση*

*Στην οικογένειά μου*

## ΠΕΡΙΛΗΨΗ

Τα τελευταία χρόνια η πιο εντυπωσιακή ανάπτυξη στη βιομηχανία της πληροφορικής προήλθε εξαιτίας της ραγδαίας ανάπτυξης του internet. Όλες οι εταιρείες και οι οργανισμοί σήμερα προσπαθούν να επωφεληθούν από αυτή την ανάπτυξη μέσω της παρουσίας τους στον παγκόσμιο ιστό είτε για την παροχή προϊόντων και υπηρεσιών (B2C Business to Customer) είτε για τη συνεργασία με άλλες επιχειρήσεις με σκοπό την παροχή σύνθετων υπηρεσιών προς τους τελικούς χρήστες (B2B Business to Business). Η συνεργασία και ο ανταγωνισμός των επιχειρήσεων σήμερα γίνεται μέσα στο ψηφιακό περιβάλλον του internet το οποίο εκμηδενίζει τις αποστάσεις, επιταχύνει και αυτοματοποιεί σε μεγάλο βαθμό τη συνεργασία και τον ανταγωνισμό. Συχνά, το περιβάλλον αυτό αποκαλείται ψηφιακό οικοσύστημα επιχειρήσεων, ένα περιβάλλον στο οποίο οι επιχειρήσεις ζουν, αναπτύσσονται και πεθαίνουν όπως σε ένα φυσικό οικοσύστημα. Ένα κρίσιμο στοιχείο σε αυτό το περιβάλλον είναι η επικοινωνία μεταξύ των επιχειρήσεων, η οποία πρέπει να είναι καταληπτή πλήρως ώστε να μπορούν να παρθούν αποφάσεις και να αυτοματοποιηθούν διαδικασίες χωρίς την ανάγκη ανθρώπινης παρεμβολής.

Η σύνταξη των μηνυμάτων στο internet σήμερα βασίζεται σε μια πρότυπη γλώσσα, την XML (eXtensible Markup Language) και τα παράγωγά της (XML Schema, XSL (eXtensible Stylesheet Language), XPath (XML Path Language), XQuery (XML Query Language) κλπ.). Ωστόσο, η σύνταξη αυτή δεν είναι αρκετή για να αποκαλύψει τη σημασιολογία των μηνυμάτων που ανταλλάσσονται. Οι οντολογίες πεδίου γνώσης (Domain Ontologies) χρησιμοποιούνται ακριβώς για το σκοπό αυτό, δηλαδή για να προσθέσουν στην τυπική σύνταξη των μηνυμάτων, τα οποία ανταλλάσσονται μεταξύ επιχειρήσεων που συνεργάζονται, σημασιολογία η οποία διευκολύνει και αυτοματοποιεί τη συνεργασία.

Στο πλαίσιο του Ευρωπαϊκού Ερευνητικού Προγράμματος DBE (Digital Business Ecosystem) δημιουργήθηκε από το εργαστήριο Διανεμημένων Πληροφοριακών Συστημάτων και Εφαρμογών (MUSIC) μια γλώσσα ορισμού οντολογιών (Ontology Definition Metamodel – ODM) χρησιμοποιώντας εργαλεία μοντελοποίησης που ορίζονται στο μοντέλο της αρχιτεκτονικής MOF (Meta Object Facility) του διεθνούς οργανισμού τυποποίησης OMG (Object Management Group).

Στην παρούσα διπλωματική διατριβή, σχεδιάστηκε και υλοποιήθηκε ένα γραφικό εργαλείο για την υποστήριξη της δημιουργίας και επεξεργασίας οντολογιών με τη χρήση σημειολογίας συμβατής με αυτή της UML (Unified Modeling Language). Η εσωτερική αναπαράσταση των οντολογιών που δημιουργεί ο χρήστης με το εργαλείο αυτό αναπαρίστανται με το ODM και αποθηκεύονται σε κατάλληλη βάση γνώσης (Knowledge

Base) συμβατή με το πρότυπο JMI (Java Metadata Interface), και μπορούν να περιγράφονται με τη χρήση της τεχνολογίας XMI (XML Metadata Interchange). Δεδομένης της συμβατότητας του μετά-μοντέλου ODM και του μοντέλου της γλώσσας OWL (Ontology Web Language), είναι δυνατόν με κατάλληλους μετασχηματισμούς οι οντολογίες που αναπτύσσονται με το εργαλείο αυτό να περιγράφονται σε σύνταξη OWL. Στο εργαλείο δόθηκε η ονομασία GRAMOFONE, που αποτελεί σύντμηση των λέξεων **GRA**phical, **MOF**-based, **ON**tology **E**ditor. Το GRAMOFONE υλοποιήθηκε πάνω στην πλατφόρμα ανάπτυξης εφαρμογών Eclipse χρησιμοποιώντας το εργαλείο (plug-in) GEF (Graphical Editing Framework) το οποίο παρέχει τη δυνατότητα δημιουργίας γραφικών εργαλείων και το οποίο στηρίζεται στο πρότυπο MVC (Model – View – Controller). Το GRAMOFONE και το ODM αξιολογήθηκαν μέσω της δημιουργίας μίας οντολογίας κρασιών γραμμένη σε OWL η οποία δίδεται από το πανεπιστήμιο Stanford ως οντολογία δοκιμών για εργαλεία σχετικά με οντολογίες (ontology editors, reasoners κλπ.).

Η παρούσα διπλωματική εργασία χρηματοδοτήθηκε στο πλαίσιο του Ευρωπαϊκού Ερευνητικού Προγράμματος DBE, στο οποίο συμμετέχει το εργαστήριο Διανεμημένων Πληροφοριακών Συστημάτων και Εφαρμογών (MUSIC), του Πολυτεχνείου Κρήτης.

## ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να εκφράσω τις ιδιαίτερες ευχαριστίες μου στον επιβλέποντα καθηγητή μου κ. Σταύρο Χριστοδουλάκη για την επίβλεψη και καθοδήγησή του στην εκπόνηση αυτής της διπλωματικής εργασίας, καθώς και για τις σημαντικές εμπειρίες που μου προσέφερε κατά την διάρκεια της εργασίας μου στο Εργαστήριο Διανεμημένων Πληροφοριακών Συστημάτων και Εφαρμογών.

Θα ήθελα επίσης να ευχαριστήσω τους καθηγητές κκ. Μανόλη Κουμπαράκη και Βασίλη Σαμολαδά για το χρόνο που αφιέρωσαν στην ανάγνωση του κειμένου και για τις εποικοδομητικές παρατηρήσεις τους.

Επίσης, οφείλω ένα ιδιαίτερο ευχαριστώ στο Νεκτάριο Γιολδάση για την συνεργασία και την αρωγή που μου προσέφερε στο σχεδιασμό της εργασίας αυτής καθώς και τις συμβουλές του σε τεχνικά θέματα.

Ένα μεγάλο ευχαριστώ σε ολόκληρη την ομάδα των συναδέλφων που εργάστηκαν πάνω στην εξέλιξη και ανάπτυξη του προγράμματος DBE και ιδιαίτερα τους Νίκο Παππά, Γιώργο Ανέστη, Θέμη Δακανάλη και Νατάσα Καραναστάση για την αρμονική μας συνύπαρξη και τη συνεργασία τους.

# ΠΕΡΙΕΧΟΜΕΝΑ

# ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

Κεφάλαιο 1

# ΕΙΣΑΓΩΓΗ

**Αναγκαιότητα**

Οι οντολογίες είναι ένα μέσο προσδιορισμού των εννοιών και όρων ενός πεδίου γνώσης με τρόπο τυπικό (formal) σχεδιασμένο για επεξεργασία από μηχανές. Η συνειδητοποίηση της ανάγκης αλλαγής της προσέγγισης της γνώσης στην σύγχρονη εποχή έχει σαν συνέπεια στην IT (Information Technology) κοινότητα την μετατόπιση του φορτίου της σύλληψης του νοήματος των δεδομένων από τις παραδοσιακές μεθόδους ανάπτυξης αλγορίθμων και κανόνων στην αναπαράσταση των ίδιων των δεδομένων.

Ανοίγοντας το International Semantic Web Conference το 2003, ο πρόεδρος του συνεδρίου Jim Hendler δήλωσε ότι "a little semantics goes a long way". Αυτό που εννοούσε ήταν πως η ενσωμάτωση έστω και μίας μικρής σημασιολογικής ποιότητας στα δεδομένα μας (τα οποία μπορεί να ανήκουν σε ιστοσελίδες, πίνακες βάσεων δεδομένων, έγγραφα σε ηλεκτρονική μορφή ή οτιδήποτε άλλο) μπορεί να έχει σαν αποτέλεσμα τα δεδομένα αυτά να γίνουν πιο άμεσα, πιο χρήσιμα, και με μεγαλύτερο εύρος δυνατοτήτων για όλες τις εφαρμογές που λαμβάνουν γνώση του πλαισίου γνώσης – αναπαράστασης, το οποίο αποτελεί η οντολογία. Για τον λόγο αυτό, υπάρχει η διαρκώς αυξανόμενη αίσθηση μεταξύ των ερευνητών και των επαγγελματιών ότι οι οντολογίες θα παίξουν ένα πολύ σημαντικό ρόλο στην επερχόμενη εξέλιξη της διαχείρισης των πληροφοριών. [3]

Από την αρχή της δεκαετίας του ενενήντα οι οντολογίες έχουν γίνει ένας δημοφιλής τομέας έρευνας από αρκετές ερευνητικές κοινότητες τεχνητής νοημοσύνης (AI – Artificial Intelligence), συμπεριλαμβανομένων των knowledge engineering, natural-language processing και knowledge representation κοινοτήτων. Πιο πρόσφατα, η έννοια της οντολογίας έχει επεκταθεί ευρέως σε τομείς όπως intelligent information integration, information retrieval στο Internet και knowledge management. Η μεγάλη δημοτικότητα των οντολογιών οφείλεται σε ένα μεγάλο ποσοστό στο τι προμηνύεται από την χρήση των οντολογιών: μία διαμοιρασμένη και κοινή κατανόηση ενός τομέα η οποία μπορεί να κοινοποιηθεί μεταξύ ανθρώπων και υπολογιστών.

Αναμένεται πως η αναγκαιότητα της χρήσης οντολογιών, και κατά συνέπεια εργαλείων τα οποία να υποστηρίζουν την δημιουργία τέτοιων οντολογιών, θα αυξηθεί. Η δημιουργία μιας οντολογίας είναι μια διαδικασία μοντελοποίησης (μοντελοποίηση των εννοιών ενός συγκεκριμένου πεδίου γνώσης καθώς και των συσχετίσεων μεταξύ των εννοιών αυτών). Ως τέτοια, η διαδικασία αυτή θα μπορούσε να γίνει πιο άμεσα κατανοητή σε χρήστες οι οποίοι είναι ήδη εξοικειωμένοι με διαδεδομένες τεχνολογίες μοντελοποίησης. Μια από τις πιο διαδεδομένες τεχνολογίες μοντελοποίησης η οποία χρησιμοποιείται από έναν πολύ μεγάλο αριθμό χρηστών (user group) είναι η γλώσσα μοντελοποίησης UML (Unified Modeling Language) [30]. Η γλώσσα έχει στην ουσία τυποποιήσει και καθιερώσει μια μεθοδολογία μοντελοποίησης και κάνει χρήση μιας συγκεκριμένης σημειολογίας (notation) η οποία είναι πλέον γνωστή σε ένα μεγάλο αριθμό χρηστών.

Ένα εργαλείο το οποίο θα δίνει τη δυνατότητα δημιουργίας οντολογιών με τρόπο παρόμοιο με αυτόν της UML θα έχει μεγάλη απήχηση και θα κάνει τη διαδικασία ανάπτυξης οντολογιών μια «γνώριμη» διαδικασία για πολλούς χρήστες. Την ανάγκη αυτή προσπαθεί να ικανοποιήσει η παρούσα διπλωματική εργασία.

**Οντολογίες**

Η λέξη "οντολογία" φαίνεται πως προκαλεί σύγχυση στους περισσότερους ανθρώπους που ασχολούνται με την αναπαράσταση της γνώσης. Ο όρος είναι δανεισμένος από την φιλοσοφία, στην οποία μία Οντολογία είναι μία συστηματική θεώρηση της Ύπαρξης. Για τα συστήματα Τεχνητής Νοημοσύνης (Artificial Intelligence (AI) systems), αυτό που "υπάρχει" είναι οτιδήποτε το οποίο μπορεί να αναπαρασταθεί. Στο πλαίσιο του διαμοιρασμού της γνώσης, ο όρος οντολογία χρησιμοποιείται σαν ο προσδιορισμός μίας γενικότερης αντίληψης. Όταν η γνώση ενός τομέα αναπαριστάται με τρόπο τυπικό και επεξηγηματικό τότε το σύνολο των αντικειμένων τα οποία μπορούν να αναπαρασταθούν ονομάζεται "universe of discourse" (τομέας γνώσης της ομιλίας). Το σύνολο αυτό των αντικειμένων, και οι μεταξύ τους σχέσεις που περιγράφονται, αναπαριστώνται χρησιμοποιώντας ένα λεξιλόγιο με το οποίο μία knowledge-based γλώσσα αναπαριστά την γνώση. Με τον τρόπο αυτό, στο πλαίσιο των κοινοτήτων Τεχνητής Νοημοσύνης, μπορούμε να περιγράψουμε μία οντολογία ενός πεδίου γνώσης προσδιορίζοντας ένα σύνολο από όρους που μπορούν να αναπαρασταθούν. Σε μια τέτοια οντολογία, οι προσδιορισμοί συσχετίζουν τα ονόματα των οντοτήτων του "universe of discourse" (π.χ. classes, relations, functions, ή άλλα αντικείμενα) μέσω κειμένου που μπορεί να διαβαστεί από ανθρώπους το οποίο περιγράφει την σημασιολογία των ονομάτων, και σχηματίζει αξιώματα που περιορίζουν την ερμηνεία και την ορθή χρήση των όρων αυτών. [22]

Οι οντολογίες χρησιμοποιούνται πολλές φορές στην περιγραφή «οντολογικών δεσμεύσεων (commitments)» για ένα σύνολο από πράκτορες (agents), ώστε να μπορούν να επικοινωνούν μεταξύ τους όταν αναφέρονται σε κάποιο πεδίο γνώσης με μεγαλύτερη ευκολία έχοντας ένα κοινό γνωστικό υπόβαθρο. Ένας πράκτορας είναι σύμφωνος (ακολουθεί) με μία οντολογία όταν οι ενέργειές του οι οποίες παρατηρούνται είναι συνεπείς με τους προσδιορισμούς που περιέχονται στην οντολογία. Μία οντολογία προσδιορίζει το λεξιλόγιο με το οποίο οι πράκτορες επικοινωνούν μεταξύ τους. Οι «οντολογικές δεσμεύσεις» είναι συμφωνίες με σκοπό τη χρήση ενός κοινού λεξιλογίου με τρόπο κατανοητό και συνεπή. Οι πράκτορες οι οποίοι διαμοιράζονται ένα λεξιλόγιο δεν είναι υποχρεωμένοι να διαμοιράζονται μία γνωστική βάση δεδομένων. Κάθε πράκτορας μπορεί να «γνωρίζει» πράγματα τα οποία οι υπόλοιποι δεν γνωρίζουν. Επιπλέον ένας πράκτορας ο οποίος δεσμεύεται από μία οντολογία δεν είναι υποχρεωμένος να απαντά σε όλα τα ερωτήματα (queries) τα οποία μπορούν να διατυπωθούν στο πλαίσιο του κοινού λεξιλογίου.

Εν συντομία, μία δέσμευση σε μία κοινή οντολογία εγγυάται την συνέπεια, αλλά όχι την πληρότητα της πληροφορίας, σε σχέση με τα ερωτήματα (queries) και τις δηλώσεις (assertions) τα οποία μπορούν να διατυπωθούν χρησιμοποιώντας το λεξιλόγιο που προσδιορίζεται στην οντολογία.

**Ψηφιακά Οικοσυστήματα για Επιχειρήσεις και το πρόγραμμα DBE**

Πολλές βιομηχανίες σήμερα συμπεριφέρονται σαν ένα μαζικά αλληλοσυνδεόμενο δίκτυο οργανισμών, τεχνολογιών, καταναλωτών και προϊόντων. Σύντομα, με τη χρήση των τεχνολογιών του ηλεκτρονικού εμπορίου, ο βαθμός της αλληλεπίδρασης μεταξύ των εταιριών στην βιομηχανία θα είναι εντυπωσιακός, και η επιτυχία μίας επιχείρησης δεν θα εξαρτάται πλέον από το μέγεθος των εσωτερικών και εξωτερικών λειτουργιών της, αλλά από τους συνδέσμους και τις σχέσεις που εγκαθιστά με άλλες επιχειρήσεις του επιχειρηματικού περιβάλλοντος.

Σε ένα τέτοιο περιβάλλον η δομή και η λειτουργία μιας επιχείρησης επηρεάζονται (μεταβάλλονται) τόσο από τις συνθήκες εκτός του επιχειρηματικού περιβάλλοντος, όσο και από τις τάσεις ή τις ανάγκες που διαμορφώνονται στην αγορά. Η επιβίωση και επιτυχία μιας επιχείρησης εξαρτάται από τη γνώση που έχει για το περιβάλλον της, και από τη δυνατότητά της να προσαρμόζεται (δομή, λειτουργία, συνεργασίες, κλπ) στις αλλαγές που συμβαίνουν γύρω της. Η λειτουργία του όλου περιβάλλοντος μοιάζει με ένα επιχειρηματικό οικοσύστημα (Business Ecosystem) όπου οι οργανισμοί που ζουν σε αυτό (επιχειρήσεις) συνεργάζονται, ανταγωνίζονται και συν-εξελίσσονται.

Με τη διάδοση των τεχνολογιών διαδικτύου και συγκεκριμένα αυτής των υπηρεσιών διαδικτύου, καθίσταται δυνατή η ανάπτυξη Ψηφιακών Επιχειρηματικών Οικοσυστημάτων (Digital Business Ecosystems). Ένα ψηφιακό επιχειρηματικό οικοσύστημα ορίζεται ως το εικονικό εκείνο περιβάλλον όπου υπάρχει ψηφιακή αναπαράσταση των επιχειρήσεων (π.χ. περιγραφή των λειτουργιών τους, των επιχειρησιακών μοντέλων τους, κλπ.), των υπηρεσιών που αυτές προσφέρουν στον έξω κόσμο (π.χ. πώληση προϊόντων, ενοικίαση δωματίων, μεταφορά δεμάτων, κλπ.), και διεκπεραιώνονται επιχειρηματικές διαπραγματεύσεις, συνεργασίες, και συνδιαλλαγές επί τη βάση αυτών των υπηρεσιών.

Το πρόγραμμα DBE (Digital Business Ecosystem) στοχεύει στην ανάπτυξη των τεχνολογιών και την ανάπτυξη της κατάλληλης υποδομής που θα καθιστά ικανή τη δημιουργία Ψηφιακών Επιχειρηματικών Οικοσυστημάτων (Digital Business Ecosystems). Συγκεκριμένα αποσκοπεί στο να δημιουργήσει ένα περιβάλλον, βασισμένο στο Internet, το οποίο θα δίνει στις εφαρμογές υπηρεσιών διαδικτύου (web services) των επιχειρήσεων την ευκολία να οργανώνονται και να βελτιώνονται με ημιαυτόματο τρόπο, μιμούμενες εξελικτικά φαινόμενα τα οποία παρατηρούνται και στον φυσικό κόσμο. [1]

**Αναπαράσταση Γνώσης στο DBE**

Βασική προϋπόθεση για την επίτευξη του σκοπού αυτού είναι η σωστή αναπαράσταση και η αξιοποίηση της γνώσης. Η γνώση αυτή μπορεί να αφορά σε γενικές έννοιες ενός επιχειρηματικού τομέα (business domain), στο επιχειρησιακό μοντέλο μιας εταιρίας και τις υπηρεσίες / προϊόντα που αυτή προσφέρει, στο νομικό πλαίσιο των αγορών, κλπ. Σημαντικό επίσης είναι η γνώση αυτή να μπορεί να αξιοποιηθεί τόσο για την επίτευξη σημασιολογικής αλληλεπίδρασης των επιχειρήσεων, όσο και για την ημι-αυτόματη «μετάφραση» αυτής της γνώσης σε λογισμικό υλοποίησης των επιχειρηματικών συστημάτων και υπηρεσιών.

Για την αναπαράσταση γνώσης η χρήση οντολογιών είναι η κυρίαρχη τάση και διάφορες γλώσσες αναπαράστασής της υπάρχουν με πιο διαδεδομένη (πλέον) τη γλώσσα OWL (Ontology Web Language). Η ανάγκη ύπαρξης οντολογιών (για επιχειρήσεις και υπηρεσίες) στο DBE έχει αναγνωριστεί από τα πρώτα στάδια ανάπτυξης του προγράμματος. Οι οντολογίες είναι γνωστοί μηχανισμοί για την αναπαράσταση της γνώσης και χρησιμοποιούνται συχνά στον προσδιορισμό κάποιων ευρύτερων αντιλήψεων και εννοιών. Οι οντολογίες έχουν τρία βασικά χαρακτηριστικά: [2]

1. **Είναι formal** (τυπικές), το οποίο σημαίνει ότι μπορούν να διαβαστούν από μηχανές.
2. **Είναι σαφείς**, ο τύπος των εννοιών που χρησιμοποιούνται και οι περιορισμοί της χρήσης τους ορίζονται με σαφήνεια.

3. **Προσδιορίζουν μία κοινή αντίληψη**, το οποίο σημαίνει ότι μία οντολογία εσωκλείει μία ομόφωνη και προφορική γνώση, δηλαδή γνώση η οποία δεν είναι υιοθετημένη από ένα άτομο αλλά από μία κοινότητα.

Στο πλαίσιο του προγράμματος DBE ο στόχος της ημι-αυτόματης παραγωγής λογισμικού από την επιχειρηματική γνώση, οδήγησε στην υιοθέτηση της τεχνολογίας MDA (Model Driven Architecture). Η τεχνολογία αυτή ορίζει μια πολύ-επίπεδη αρχιτεκτονική δόμησης γνώσης, όπου στο πιο χαμηλό επίπεδο υπάρχουν μοντέλα λογισμικού ενώ στο πιο πάνω επίπεδο υπάρχουν τα μοντέλα περιγραφής της επιχειρηματικής λογικής που το λογισμικό αυτό υλοποιεί. Ο θεμελιώδης μηχανισμός μοντελοποίησης στο πλαίσιο του MDA είναι το MOF (Meta Object Facility) το οποίο είναι η γλώσσα περιγραφής μετά-μοντέλων, δηλαδή γλωσσών μοντελοποίησης. Για το λόγο αυτό, η αναπαράσταση οντολογιών στο DBE θα πρέπει να βασίζεται στην τεχνολογία MOF προκειμένου η γνώση που αυτές περιγράφουν να μπορεί να μεταφραστεί με ημιαυτόματο τρόπο και να ενσωματωθεί σε συστήματα λογισμικού που υλοποιούν σχετικές επιχειρηματικές υπηρεσίες.
Όπως ειπώθηκε, για το σκοπό αυτό στο DBE έχει αναπτυχθεί, με τη χρήση του MOF, ένα μετά-μοντέλο ορισμού οντολογιών (ODM: Ontology Definition Metamodel) το οποίο είναι συμβατό με τη γλώσσα OWL και επίσης κινείται στην ίδια κατεύθυνση με τον οργανισμό OMG ο οποίος βρίσκεται σε διαδικασία υιοθέτησης ενός αντίστοιχου μετά-μοντέλου για την περιγραφή οντολογιών.
Η αρχιτεκτονική προσέγγιση που ακολουθείται στο πλαίσιο του DBE και σε σχέση με τις οντολογίες είναι η εξής: Οι ειδικοί ενός επιχειρηματικού τομέα δημιουργούν με τη χρήση του GRAMOFONE οντολογίες που περιγράφουν την κοινά αποδεκτή σημασιολογία του συγκεκριμένου τομέα. Οι οντολογίες αυτές ονομάζονται domain specific ontologies, και σαν παράδειγμα, κάποιος μπορεί να παραθέσει οντολογίες για τον τομέα τουρισμού, τον τομέα τηλεπικοινωνιών κτλ. Οι οντολογίες αυτές χρησιμοποιούνται από τις διάφορες επιχειρήσεις του οικοσυστήματος για την περιγραφή των επιχειρηματικών τους μοντέλων και των υπηρεσιών που αυτές διαθέτουν [2].

**Στόχοι της Εργασίας**

Η εργασία αυτή πραγματοποιήθηκε στο πλαίσιο του προγράμματος DBE και έχει σαν κεντρικό θέμα της τον σχεδιασμό και την υλοποίηση ενός γραφικού εργαλείου το οποίο να υποστηρίζει την γραφική δημιουργία και διαχείριση οντολογιών όπως αυτές ορίζονται στο Ontology Definition Metamodel (ODM) με την χρήση σημειολογίας (notation) ομοίου με αυτού της γλώσσας UML. Συγκεκριμένα οι στόχοι της παρούσας εργασίας είναι:

- Η μελέτη του μετά-μοντέλου ODM, και ο προσδιορισμός των απαιτήσεων που απορρέουν από αυτό για την ανάπτυξη ενός εργαλείου επεξεργασίας οντολογιών.
- Ο σχεδιασμός και υλοποίηση ενός εργαλείου το οποίο θα επιτρέπει στον χρήστη την δημιουργία οντολογιών με γραφικό τρόπο οι οποίες θα ακολουθούν το ODM.
- Η χρήση σημειολογίας όμοιας με αυτή που χρησιμοποιείται από τη γλώσσα UML, η οποία είναι ευρέως γνωστή, αποτελεί πρότυπο και επιπλέον χρησιμοποιείται σήμερα από όλες τις επιχειρήσεις, ιδιαίτερα για το σχεδιασμό και την κατασκευή μεγάλων συστημάτων.
- Η υλοποίηση μηχανισμού αποθήκευσης οντολογιών στην βάση δεδομένων του DBE (Knowledge Base) καθώς επίσης και στο τοπικό σύστημα αρχείων.
- Η εξαγωγή των υλοποιημένων οντολογιών σε μορφή εγγράφων XMI 1.2.
- Η απόκρυψη, όσο γίνεται, της πολυπλοκότητας του μετά-μοντέλου, ODM, από τον χρήστη.
- Η ανάπτυξη ενός εύχρηστου Περιβάλλοντος Εργασίας (User Interface) το οποίο θα δίνει τη δυνατότητα δημιουργίας οντολογιών με το μικρότερο δυνατό κόπο από τον χρήστη.
- Η χρήση βοηθητικών λειτουργιών και αυτό-επεξηγηματικών όρων έτσι ώστε ο χρήστης να εξοικειώνεται όσο το δυνατόν συντομότερα και με μεγαλύτερη ευκολία με το περιβάλλον εργασίας.

**Δομή της Εργασίας**

Μετά το τρέχον –πρώτο– κεφάλαιο, το οποίο αποτελεί και την εισαγωγή της διπλωματικής διατριβής, ακολουθούν οκτώ ακόμα κεφάλαια.

Στο δεύτερο κεφάλαιο παρουσιάζεται η σχετική με τη διπλωματική διατριβή έρευνα, δηλαδή όλες εκείνες οι γνώσεις και τεχνολογίες οι οποίες σχετίζονται με την τεχνολογία των οντολογιών. Στο ίδιο κεφάλαιο παρουσιάζονται οι πιο δημοφιλείς εργασίες οι οποίες έχουν παρόμοιο αντικείμενο μελέτης με την παρούσα εργασία. Στο τέλος του κεφαλαίου παρατίθεται ένας συγκριτικός πίνακας των εργασιών αυτών συμπεριλαμβανομένης και της παρούσας.

Στο τρίτο κεφάλαιο περιγράφονται οι τεχνολογίες υλοποίησης που χρησιμοποιήθηκαν για τη δημιουργία του εργαλείου (GRAMOFONE).

Στο τέταρτο κεφάλαιο περιγράφονται η αρχιτεκτονική μετά-δεδομένων MOF (Meta Object Facility) και το μετά-μοντέλο ορισμού οντολογιών ODM (Ontology Definition Metamodel).

Στο πέμπτο κεφάλαιο αναλύονται οι απαιτήσεις του συστήματος με την καταγραφή όλων των δυνατών σεναρίων αλληλεπίδρασης του χρήστη με το εργαλείο, GRAMOFONE. Η καταγραφή αυτή πραγματοποιείται με την χρήση της τεχνολογίας ανάλυσης απαιτήσεων των Use Cases. [21]

Στο έκτο κεφάλαιο περιγράφεται η αρχιτεκτονική του συστήματος.

Στο έβδομο κεφάλαιο περιγράφεται η υλοποίηση του συστήματος και ακολουθεί αναλυτική περιγραφή των δυνατοτήτων του.

Στο όγδοο κεφάλαιο παρουσιάζονται τα αποτελέσματα της αξιολόγησης του μετά-μοντέλου ODM, με βάση την εμπειρία υλοποίησης του εργαλείου αλλά και από την ανάπτυξη μίας πρότυπης οντολογίας (Wine Ontology).

Στο ένατο κεφάλαιο γίνεται μία ανακεφαλαίωση, αναφέρονται τα συμπεράσματα της εργασίας που παρουσιάζεται στο κείμενο αυτό, και γίνονται κάποιες επισημάνσεις για μελλοντικές επεκτάσεις του εργαλείου GRAMOFONE.

**Ανακεφαλαίωση**

Σε αυτό το κεφάλαιο τεκμηριώθηκε η αναγκαιότητα της παρούσας διπλωματικής διατριβής, περιγράφηκε η τεχνολογία των οντολογιών, έγινε μία σύντομη περιγραφή των Ψηφιακών Οικοσυστημάτων Επιχειρήσεων (Digital Business Ecosystems) και του ευρωπαϊκού ερευνητικού προγράμματος DBE στο πλαίσιο του οποίου αναπτύχθηκε η παρούσα διπλωματική διατριβή, περιγράφηκε η αναπαράσταση της γνώσης στο πρόγραμμα DBE, παρουσιάστηκαν οι στόχοι της εργασίας και τέλος δόθηκε μία σύντομη περιγραφή της δομής της παρούσας εργασίας.

Κεφάλαιο 2

# ΣΧΕΤΙΚΗ ΕΡΕΥΝΑ

## Εισαγωγή

Σε αυτό το κεφάλαιο αρχικά θα παρουσιαστούν οι XML γλώσσες περιγραφής οντολογιών RDF, RDF Schema και OWL της Semantic Web Community, οι οποίες είναι οι πιο γνωστές τεχνολογίες αναπαράστασης της γνώσης σήμερα. Στη συνέχεια θα παρουσιαστεί η γλώσσα UML, η οποία αποτελεί την πιο δημοφιλή γλώσσα μοντελοποίησης σήμερα. Τέλος θα παρουσιαστούν εργαλεία που έχουν σχετιζόμενο αντικείμενο μελέτης με το αντικείμενο μελέτης της παρούσας διατριβής, δηλαδή αλληλεπιδραστικά εργαλεία για την ανάπτυξη οντολογιών. Κανένα από τα εργαλεία αυτά δεν έχει αναπτυχθεί πάνω στην αρχιτεκτονική MOF (Meta Object Facility) του διεθνούς οργανισμού τυποποίησης OMG (Object Management Group), ούτε φυσικά υποστηρίζουν το Μετά-μοντέλο Ορισμού Οντολογιών (ODM). Λόγω της πληθώρας των εργασιών που έχουν γίνει πάνω στην δημιουργία εργαλείων για την επεξεργασία οντολογιών θα γίνει σύγκριση μόνο με τα πιο δημοφιλή από αυτά και κυρίως με εργαλεία τα οποία παρουσιάζουν παρόμοιες ιδιότητες με το GRAMOFONE. Αρχικά θα παρουσιαστούν τα κύρια χαρακτηριστικά κάθε εργαλείου και στο τέλος θα γίνει η μεταξύ τους σύγκριση. Η μελέτη που παρουσιάζεται στην ενότητα αυτή, βασίζεται σε μεγάλο βαθμό στην εργασία του Michael Deny, Ontology Tools Survey Revisited [3].

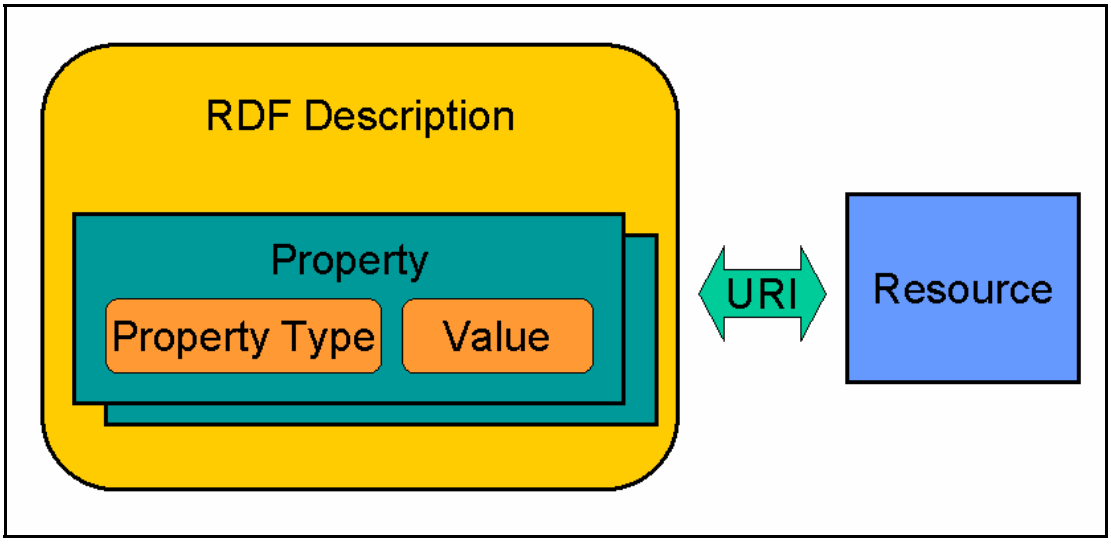
## RDF (Resource Description Framework)

Η γλώσσα RDF, η οποία αναπτύχθηκε από το W3C, παρέχει τα θεμέλια της διαχείρισης μετά-δεδομένων ανάμεσα σε διαφορετικές κοινότητες περιγραφής πηγών πληροφοριών. Ένα από τα σημαντικότερα εμπόδια που αντιμετώπιζαν οι κοινότητες αυτές ήταν ο μεγάλος αριθμός των μη συμβατών προτύπων (standards) που αφορούσαν την σύνταξη και τον προσδιορισμό γλωσσών περιγραφής μετά-δεδομένων και των σχημάτων (schemas) τους. Αυτό είχε οδηγήσει στην ελλιπή ανάπτυξη εφαρμογών και υπηρεσιών διαχείρισης μετά-δεδομένων ανάμεσα στις κοινότητες περιγραφής πηγών πληροφοριών. Η RDF γλώσσα

παρέχει την λύση σε αυτά τα προβλήματα μέσω του προσδιορισμού της σύνταξής της (RDF specification, W3C, 1999a) και του προσδιορισμού του σχήματος Resource Description Framework Schema (RDF Schema specification, W3C, 1998a). [23]

Σκοπός της γλώσσας RDF είναι η υποστήριξη της ενιαίας διαχείρισης και χρήσης των μετά-δεδομένων. Η RDF καθιστά δυνατή την περιγραφή πηγών πληροφοριών στο Διαδίκτυο (οποιοδήποτε αντικείμενο με ένα Uniform Resource Identifier (URI) ως διεύθυνση) με μορφή που μπορεί να γίνει κατανοητή από μηχανές. Το γεγονός αυτό καθιστά δυνατή την αναπαράσταση και την εκμετάλλευση της σημασιολογίας των αντικειμένων. Με την ευρεία ανάπτυξη της RDF, θα είναι δυνατή η ανάπτυξη, από τις υπηρεσίες, κανόνων για την αυτοματοποίηση των λειτουργιών διαχείρισης της πληροφορίας στο Διαδίκτυο.

Η RDF βασίζεται σε ένα συμπαγές και τυπικό μοντέλο το οποίο χρησιμοποιεί κατευθυνόμενους γράφους οι οποίοι κρύβουν την σημασιολογία της πληροφορίας που περιγράφουν. Η βασική αρχή είναι πως μία πηγή πληροφορίας (Resource) περιγράφεται μέσω ενός συνόλου από Ιδιότητες (Properties) που ονομάζονται RDF Περιγραφή (Description). Κάθε μία από αυτές τις Ιδιότητες έχει έναν Τύπο (Property Type) και μία Τιμή (Property Value). Οποιαδήποτε πηγή πληροφορίας μπορεί να περιγραφεί με την RDF εφόσον η πληροφορία αυτή αναγνωρίζεται με ένα URI όπως φαίνεται στην παρακάτω εικόνα.




**Εικόνα 1: Resource Description Framework Μοντέλο**

Η RDF βασίζεται στην XML και χρησιμοποιεί την λειτουργία των Namespaces της XML. Το XML Namespace, το οποίο «δείχνει» σε ένα URI, επιτρέπει στο RDF να θέσει το πεδίο ορισμού και να προσδιορίσει μοναδικά ένα σύνολο από ιδιότητες. Αυτό το σύνολο των ιδιοτήτων, το οποίο ονομάζεται σχήμα (schema), είναι προσβάσιμο στο URI που ορίζεται

στο namespace. Ένα παράδειγμα RDF σύνταξης η οποία περιγράφει μία αναφορά φαίνεται στην παρακάτω εικόνα.

```
<? xml version="1.0" ?>
<RDF xmlns = "http://w3.org/TR/1999/PR-rdf-syntax-19990105#"
    xmlns:DC = "http://purl.org/DC#" >

  <Description about = "http://dstc.com.au/report.html" >
    <DC:Title> The Future of Metadata </DC:Title>
    <DC:Creator> Jacky Crystal </DC:Creator>
    <DC:Date> 1998-01-01 </DC:Date>
    <DC:Subject> Metadata, RDF, Dublin Core </DC:Subject>
  </Description>
</RDF>
```

**Εικόνα 2: Παράδειγμα RDF Σύνταξης**

Στην πρώτη γραμμή ορίζεται πως πρόκειται για ένα XML έγγραφο. Στην επόμενη γραμμή ορίζονται δύο namespaces. Το πρώτο namespace (αυτό που δεν περιλαμβάνει πρόθεμα) δηλώνεται σαν το προεπιλεγμένο namespace. Για να χρησιμοποιηθούν οι ιδιότητες που ορίζονται από το δεύτερο namespace, το οποίο χρησιμοποιεί το πρόθεμα «DC» (Dublin Core), θα πρέπει να περιλαμβάνουν το πρόθεμα αυτό στη δήλωσή τους (π.χ. DC:Title). Όλες οι ιδιότητες στην περιγραφή θα προέρχονται από ένα από τα δύο namespaces. Στο κύριο τμήμα του παραδείγματος (μεταξύ των ετικετών <Description> και </Description>) ορίζονται τέσσερις Ιδιότητες οι οποίες περιγράφουν την πληροφορία την οποία «δείχνει» το URI του χαρακτηριστικού (attribute) «about» της ετικέτας <Description>. Αυτές οι ιδιότητες προέρχονται από το namespace Dublin Core (DC). Σε αυτή την περίπτωση οι ιδιότητες Τίτλος (Title), Δημιουργός (Creator), Ημερομηνία (Date), και Θέμα (Subject) δηλώνονται για την πληροφορία (resource) που είναι διαθέσιμη στην διεύθυνση <http://dstc.com.au/report.html>.

### RDFS (Resource Description Framework Schema)

Η RDF επίσης περιλαμβάνει ένα μηχανισμό προσδιορισμού σχημάτων (schemas) μετά-δεδομένων. Ένα σχήμα ορίζει το νόημα, τα χαρακτηριστικά, και τις σχέσεις ενός συνόλου ιδιοτήτων. Επιπλέον μπορεί να περιλαμβάνει περιορισμούς σε πιθανές τιμές ή περιορισμούς στην κληρονομικότητα των ιδιοτήτων από άλλα σχήματα. Το RDF Schema βασίζεται στο ίδιο μοντέλο που στηρίζεται και η RDF. Το παρακάτω παράδειγμα παρουσιάζει ένα RDF Schema στο οποίο ορίζονται δύο ιδιότητες (Title και Creator). Κάθε μία ιδιότητα ορίζεται σαν τύπος «property» μέσω μιας ετικέτας και μιας μικρής περιγραφής. Για παράδειγμα η ιδιότητα «Title» ορίζεται πως είναι τύπου (type) «Property», έχει ετικέτα (label) «Title», και συνοδεύεται από ένα σχόλιο (comment). Ένα XML Namespace URI είναι σκόπιμο να «δείχνει» σε ένα RDF Schema. Στην συγκεκριμένη περίπτωση το παρακάτω παράδειγμα

μπορεί να βρίσκεται στην διεύθυνση <http://purl.org/DC/> το οποίο θα γίνει σε αυτή την περίπτωση το URI για το namespace «DC».

```
<? xml version="1.0" ?>
<RDF  xmlns = "http://w3.org/TR/PR-rdf-syntax#"
    xmlns:RDFS = "http://w3.org/TR/WD-rdf-schema#" >
 <Description ID = "Title" >
  <type resource = "http://w3.org/TR/PR-rdf-syntax#Property" />
  <RDFS:label> Title  </RDFS:label>
  <RDFS:comment> The name given to the resource, usually by the
          Creator or Publisher </RDFS:comment>
 </Description>
 <Description ID = "Creator" >
  <type resource = "http://w3.org/TR/PR-rdf-syntax#Property" />
  <RDFS:label> Author or Creator  </RDFS:label>
  <RDFS:comment> The person or organisation primarily responsible
          for the intellectual content of the resource
         </RDFS:comment>
 </Description>
</RDF>
```

**Εικόνα 3: Παράδειγμα RDF Schema**

### Γλώσσα Περιγραφής Οντολογιών Ιστού - Ontology Web Language (OWL)

Η γλώσσα περιγραφής οντολογιών ιστού (Ontology Web Language / στο εξής OWL) έχει υιοθετηθεί από το W3C (World Wide Web Consortium) ως η πρότυπη γλώσσα περιγραφής οντολογιών. Η OWL προορίζεται να χρησιμοποιηθεί στις περιπτώσεις κατά τις οποίες η πληροφορία που περιέχεται σε έγγραφα χρειάζεται να επεξεργαστεί από εφαρμογές, σε αντίθεση με τις περιπτώσεις κατά τις οποίες το περιεχόμενο χρειάζεται μόνο να παρουσιαστεί στους ανθρώπους. Η OWL χρησιμοποιείται για την περιγραφή των εννοιών ενός πεδίου γνώσης καθώς και των σχέσεων μεταξύ των εννοιών αυτών. Αυτή η αναπαράσταση των εννοιών και των εσωτερικών τους σχέσεων ονομάζεται οντολογία. Η OWL έχει περισσότερες διευκολύνσεις για την αναπαράσταση νοημάτων και εννοιών από γλώσσες όπως η XML, η RDF, και η RDF-S, και για τον λόγο αυτό έχει υιοθετηθεί ως η πρότυπη τεχνολογία περιγραφής οντολογιών. Η OWL αποτελεί την αναθεωρημένη έκδοση της DAML + OIL γλώσσας περιγραφής οντολογιών στο Διαδίκτυο ενσωματώνοντας συμπεράσματα τα οποία προέκυψαν από τον σχεδιασμό και την εφαρμογή της DAML + OIL γλώσσας. [14]

Η OWL έχει σχεδιαστεί για την ικανοποίηση της ανάγκης δημιουργίας μίας Web γλώσσας οντολογιών. Προσθέτει περισσότερο λεξιλόγιο (από τις XML, XML Schema, RDF και RDF Schema) για την περιγραφή ιδιοτήτων (properties), και κλάσεων (classes): μεταξύ άλλων, σχέσεων μεταξύ κλάσεων (π.χ. disjointness), περιορισμών πολλαπλότητας (cardinality constraints) (π.χ. "ακριβώς ένα"), ισοδυναμιών μεταξύ κλάσεων, χαρακτηριστικών των ιδιοτήτων (π.χ. συμμετρία) κτλ.

Η OWL παρέχεται σε τρεις διαφορετικές εκδόσεις και κάθε έκδοση παρέχει μεγαλύτερη εκφραστικότητα σε σχέση με την προηγούμενη. Οι εκδόσεις αυτές είναι:

i. Η OWL Lite η οποία υποστηρίζει τους χρήστες που κατά κύριο λόγο χρειάζονται μία ιεραρχία ταξινομήσεων και απλών περιορισμών. Για παράδειγμα, ενώ υποστηρίζει περιορισμούς πολλαπλότητας (cardinality constraints), επιτρέπει μόνο το 0 ή το 1 σαν τιμές των περιορισμών αυτών. Η έκδοση αυτή παρέχει ένα γρήγορο μηχανισμό μεταφοράς γνώσης από παραδοσιακούς θησαυρούς και άλλου είδους ιεραρχικές δομές ταξινόμησης. Η υποστήριξη της έκδοσης αυτής από κάποιο εργαλείο είναι πιο εύκολη από ότι η υποστήριξη για πιο πολύπλοκες συγγενικές της γλώσσες.

ii. Η OWL DL η οποία υποστηρίζει τους χρήστες που θέλουν την μέγιστη εκφραστικότητα σε συνδυασμό με την διατήρηση της υπολογιστικής πληρότητας (όλα τα συμπεράσματα είναι εγγυημένα υπολογίσιμα) και της αποφασιστικότητας (decidability)(όλοι οι υπολογισμοί θα τελειώσουν σε πεπερασμένο χρόνο). Η έκδοση αυτή περιλαμβάνει όλες τις γλωσσικές δομές της OWL, αλλά η χρήση τους μπορεί να γίνει μόνο κάτω από συγκεκριμένους περιορισμούς (για παράδειγμα, ενώ μία κλάση μπορεί να είναι υπό-κλάση πολλών κλάσεων, μία κλάση δεν μπορεί να είναι στιγμιότυπο μίας άλλης κλάσης). Η OWL DL ονομάζεται έτσι λόγω της αντιστοιχίας της με την περιγραφική λογική (description logics), έναν τομέα έρευνας ο οποίος μελετά τη σημειολογία (logics) που σχηματίζει το θεμέλιο της OWL.

iii. Η OWL Full η οποία προορίζεται για τους χρήστες που θέλουν το μέγιστο της εκφραστικότητας και την συντακτική ελευθερία της RDF χωρίς υπολογιστικές εγγυήσεις. Για παράδειγμα, στην OWL Full μία κλάση μπορεί να θεωρηθεί ταυτόχρονα σαν μία συλλογή από άτομα (individuals) αλλά και σαν ένα individual με τα δικά του χαρακτηριστικά. Η OWL Full επιτρέπει σε μία οντολογία να επεκτείνει το νόημα ενός προϋπάρχοντος (RDF ή OWL) λεξιλογίου. Είναι απίθανο κάποιο εργαλείο συλλογιστικής (reasoning software) να είναι σε θέση να υποστηρίξει πλήρως όλα τα χαρακτηριστικά της OWL Full.

Κάθε μία από τις παραπάνω εκδόσεις της γλώσσας είναι μία επέκταση της προηγούμενης όσον αφορά στο τι είναι σε θέση να εκφραστεί νόμιμα και στο ποια συμπεράσματα μπορούν να προκύψουν με εγκυρότητα. Οι δημιουργοί οντολογιών που υιοθετούν την γλώσσα OWL θα πρέπει να κρίνουν ποια ακριβώς έκδοση της γλώσσας καλύπτει πληρέστερα τις ανάγκες τους.

Στην παρακάτω εικόνα παρουσιάζεται ένα τμήμα κώδικα OWL.

```
<owl:Class rdf:ID="WineColor">
 <owl:equivalentClass>
  <owl:Class>
   <owl:oneOf rdf:parseType="Collection">
    <Color rdf:about="#Red"/>
    <Color rdf:about="#Rose"/>
    <Color rdf:about="#White"/>
   </owl:oneOf>
  </owl:Class>
 </owl:equivalentClass>
<owl:Class>
```

**Εικόνα 4: Παράδειγμα OWL σε RDF/XML σύνταξη**

Στο παράδειγμα ορίζεται η κλάση «WineColor» σαν ισοδύναμη μίας ανώνυμης κλάσης η οποία με την σειρά της ορίζεται σαν μία συλλογή από τρία Individuals. Το χαρακτηριστικό (attribute) «rdf:about» του παραδείγματος χρησιμοποιείται στον ορισμό παραπομπών (references) σε πληροφορίες (resources) που έχουν οριστεί σε κάποιο άλλο σημείο του εγγράφου. Η σύνταξη του «rdf:about» είναι της μορφής «rdf:about=”#RESOURCE_ID”». Για παράδειγμα το πρώτο Individual που αναφέρεται στη γραμμή 5, αναφέρεται σαν Individual της κλάσης «Color» με αναγνωριστικό (id) «Red». Ουσιαστικά αυτό που ορίζεται στο παράδειγμα είναι πως το χρώμα ενός κρασιού μπορεί να είναι κόκκινο, άσπρο ή ροζέ.

**Ενοποιημένη Γλώσσα Μοντελοποίησης - Unified Modeling Language (UML)**

Οι γλώσσες RDF, RDF Schema και OWL που αναλύθηκαν προηγουμένως προέρχονται από τις κοινότητες Τεχνητής Νοημοσύνης και έχουν δομηθεί με τεχνικές προερχόμενες από αυτές τις κοινότητες. Ωστόσο οι προαναφερόμενες τεχνικές που χρησιμοποιούνται στον προσδιορισμό της γνώσης σχετίζονται περισσότερο με τις κοινότητες Τεχνητής Νοημοσύνης και είναι λιγότερο γνωστές στην ευρύτερη software engineering κοινότητα. Με σκοπό την γεφύρωση του κενού μεταξύ των κοινοτήτων Τεχνητής Νοημοσύνης και των software engineering κοινοτήτων, υπήρχε αρχικά στην DBE κοινότητα η σκέψη της χρήσης της UML για την περιγραφή των οντολογιών.

Η UML (Unified Modeling Language – Ενοποιημένη Γλώσσα Μοντελοποίησης) είναι ο απόγονος των αντικειμενοστρεφών μεθόδων ανάλυσης και σχεδιασμού που εμφανίστηκαν στα τέλη της δεκαετίες του '80 και στις αρχές της δεκαετίας του '90. Ενοποιεί με άμεσο τρόπο τις μεθόδους των Booch [25], Rumbaugh (OMT) [26], και Jacobson [27], αλλά η επιρροή της είναι ακόμη ευρύτερη. Η UML πέρασε από μια διαδικασία προτυποποίησης από τον οργανισμό OMG (Object Management Group) και αποτελεί τώρα πρότυπο το οποίο είναι ευρέως αποδεκτό. [15]

Η UML είναι γλώσσα μοντελοποίησης και όχι μέθοδος. Οι περισσότερες μέθοδοι αποτελούνται, τουλάχιστον θεωρητικά, από μια γλώσσα μοντελοποίησης αλλά και από μια διαδικασία. Η γλώσσα μοντελοποίησης είναι ο (κυρίως γραφικός) συμβολισμός (ή

σημειολογία) που οι μέθοδοι χρησιμοποιούν ως μέσο έκφρασης των σχεδίων. Η διαδικασία είναι ο συμβολισμός της μεθόδου για τα βήματα που πρέπει να ακολουθηθούν στην υλοποίηση ενός σχεδίου.

Το τμήμα της διαδικασίας σε πολλά βιβλία μεθόδων είναι μάλλον πρόχειρο. Πέρα από αυτό, θεωρούμε πως, όταν λένε οι περισσότεροι ότι χρησιμοποιούν μια μέθοδο, χρησιμοποιούν τη γλώσσα μοντελοποίησης, αλλά σπάνια ακολουθούν τη διαδικασία. Έτσι από πολλές απόψεις, η γλώσσα μοντελοποίησης είναι το πιο σημαντικό μέρος μιας μεθόδου. Είναι οπωσδήποτε το κλειδί για την επικοινωνία.

Δυστυχώς η UML από μόνη της δεν ικανοποιεί τις ανάγκες αναπαράστασης όρων οντολογιών οι οποίοι είναι δανεισμένοι από την Περιγραφική Λογική (Descriptive Logic), και οι οποίοι περιλαμβάνονται στις γλώσσες οντολογιών του σημασιολογικού ιστού (Semantic Web), όπως είναι η RDF, η RDF Schema και η OWL. Ωστόσο, η γλώσσα UML εκτός από την μοντελοποίηση συστημάτων (system modeling), χρησιμοποιείται και για την μοντελοποίηση εννοιών (conceptual modeling). Συνεπώς υπάρχει άμεση σχέση της γλώσσας UML με την ανάπτυξη οντολογιών καθώς η ανάπτυξη οντολογιών αποτελεί μία διαδικασία μοντελοποίησης εννοιών. Επιπλέον η UML γλώσσα είναι μία πολύ δημοφιλής γλώσσα η οποία έχει μεγάλη βάση χρηστών, γεγονός που θέλουμε να εκμεταλλευτούμε υλοποιώντας ένα εργαλείο δημιουργίας οντολογιών το οποίο να χρησιμοποιεί σημειολογία παρόμοια με την UML όπου αυτό είναι δυνατό (π.χ. με την χρήση διαγραμμάτων κλάσεων). Αυτή η κατεύθυνση ακολουθήθηκε στον σχεδιασμό και στην υλοποίηση του εργαλείου της διατριβής (GRAMOFONE ).

**Σχετικές Εργασίες**

Στη συνέχεια θα παρουσιαστούν εργαλεία που έχουν σχετιζόμενο αντικείμενο μελέτης με το αντικείμενο μελέτης της παρούσας διατριβής, δηλαδή αλληλεπιδραστικά εργαλεία για την ανάπτυξη οντολογιών.

**Protégé** (Stanford Medical Informatics, Stanford University)

Το Protégé είναι πιθανώς το δημοφιλέστερο και πιο ευρέως χρησιμοποιούμενο εργαλείο δημιουργίας και επεξεργασίας οντολογιών. Είναι ένα εργαλείο το οποίο επιτρέπει στους χρήστες τη δημιουργία οντολογιών για συγκεκριμένους τομείς (domains), τον ορισμό προσαρμοσμένων στις ανάγκες των χρηστών φορμών (forms) εισαγωγής δεδομένων, και την εισαγωγή των ίδιων των δεδομένων. Είναι επίσης μία πλατφόρμα η οποία μπορεί εύκολα να επεκταθεί μέσω plug-ins ώστε να περιλαμβάνει γραφικά components όπως διαγράμματα (graphs) και πίνακες (tables), πολύ-μέσα όπως ήχοι, εικόνες, και video, καθώς και διάφορα

format αποθήκευσης όπως η OWL (Ontology Web Language), η RDF (Resource Description Framework), η XML (Extensible Markup Language) και η HTML (Hyper Text Markup Language) [5]. Στην βασική έκδοση του Protégé πρέπει να προστεθούν δύο ακόμα plug-ins τα οποία θα του προσδώσουν παρόμοια λειτουργικότητα με την παρούσα διατριβή. Τα plug-ins αυτά είναι το Protégé Owl Plug-in και το ezOWL.

**Protégé Owl Plug-in** (Stanford Medical Informatics, Stanford University)
Το συγκεκριμένο plug-in παρέχει στο Protégé, μεταξύ άλλων, την δυνατότητα δημιουργίας και επεξεργασίας OWL οντολογιών. Το περιβάλλον εργασίας (user interface) του συγκεκριμένου εργαλείου βασίζεται στην χρήση φορμών (form-based) για την δημιουργία και επεξεργασία των οντολογιών. Το plug-in επιτρέπει στους χρήστες τα παρακάτω:

- Άνοιγμα και αποθήκευση OWL και RDF οντολογιών.
- Επεξεργασία και απεικόνιση classes, properties και SWRL κανόνων. (SWRL: Semantic Web Rule Language, η οποία είναι μία γλώσσα που συνδυάζει τις υπο-γλώσσες της OWL, OWL DL και OWL Lite, με τις Unary / Binary Datalog RuleML υπο-γλώσσες της Rule Markup Language)
- Προσδιορισμό λογικών (logical) χαρακτηριστικών κλάσεων όπως είναι οι OWL εκφράσεις.
- Χρήση reasoners όπως είναι οι description logic classifiers. [6]

**Protégé ezOWL** (ETRI - Electronics and Telecommunications Research Institute, South Korea)
Το ezOWL αποτελεί ένα plug-in για το Protégé το οποίο προσδίδει την δυνατότητα της γραφικής δημιουργίας και επεξεργασίας OWL οντολογιών στο Protégé εργαλείο [7]. Παρόλη την αδιαμφισβήτητη λειτουργικότητα που παρέχει το συγκεκριμένο plug-in στο Protégé, το βασικό του μειονέκτημα συγκριτικά με το εργαλείο που αναπτύχθηκε στην παρούσα διατριβή είναι η ύπαρξη ενός μοναδικού διαγράμματος στο οποίο μπορεί να αναπτυχθεί η οντολογία από τον χρήστη. Είναι πολύ εύκολο να φανταστεί κάποιος το μέγεθος και την πολυπλοκότητα που θα αποκτήσει το διάγραμμα αυτό όταν αναπτυχθεί πάνω του μία πολύ μεγάλη οντολογία. Η κατανόηση των όρων και των εννοιών που θα έχουν οριστεί θα είναι εξαιρετικά δύσκολη. Από την άλλη πλευρά το εργαλείο της παρούσας διατριβής (GRAMOFONE) επιτρέπει την δημιουργία ενός απεριορίστου αριθμού διαγραμμάτων τα οποία ο χρήστης θα είναι σε θέση να τα επεξεργαστεί και να τα κατανοήσει πολύ πιο εύκολα.

**Construct** (Network Inference)

Το εργαλείο Construct λειτουργεί σαν plug-in του Microsoft™ Visio και αποτελεί έναν γραφικό σχεδιαστή ο οποίος καθιστά δυνατή την γρήγορη και εύκολη δημιουργία οντολογιών (σημασιολογικά μοντέλα). Οι χρήστες μπορούν να συνεργαστούν και να διαμοιράσουν τις γνώσεις τους, και αν επιθυμούν να χρησιμοποιήσουν το Cerebra Server™ (ένας Description Logic μηχανισμός ανάλυσης και ελέγχου δεδομένων ο οποίος υποστηρίζει και reasoners) για την δημιουργία queries, για τον έλεγχο και την δοκιμή των μοντέλων τους. Επιπρόσθετα της βασικής λειτουργικότητας του Microsoft™ Visio, το Construct™ περιλαμβάνει μία ποικιλία λειτουργιών με σκοπό την αποδοτικότερη δημιουργία και διατήρηση των οντολογιών, όπως είναι ο έλεγχος για την ύπαρξη λαθών ή ασυνεπειών (error and consistency checks) κατά την δημιουργία των οντολογιών [8]. Το εργαλείο δίνει τη δυνατότητα χρήσης πολλαπλών διαγραμμάτων κατά τη διάρκεια ανάπτυξης μίας οντολογίας από το χρήστη και χρησιμοποιεί UML σημειολογία (notation) για τη δημιουργία οντολογιών. Το βασικό μειονέκτημα του συγκεκριμένου εργαλείου είναι το γεγονός πως πρόκειται για εμπορικό προϊόν και κατά συνέπεια το GRAMOFONE υπερτερεί σημαντικά καθώς είναι ένα open-source εργαλείο με δυνατότητες εξίσου σημαντικές με το Construct.

**IsaViz** (W3C - World Wide Web Consortium)

Το IsaViz είναι ένα γραφικό περιβάλλον με σκοπό το browsing και την δημιουργία RDF (Resource Description Framework) μοντέλων τα οποία αναπαριστώνται σε διαγράμματα. Περιλαμβάνει:

- Ένα 2.5D (2.5 διαστάσεων) περιβάλλον εργασίας, επιτρέποντας την ομαλή εστίαση (zoom) και πλοήγηση μέσα στον γράφο.
- Την δημιουργία και την επεξεργασία γράφων με την χρήση ελλείψεων, ορθογώνιων κουτιών και βελών.
- Την εισαγωγή RDF, / XML, Notation 3 και N-Triple γλωσσών.
- Την εξαγωγή σε RDF / XML, Notation 3, N-Triple, SVG (Scalable Vector Graphics) και PNG (Portable Network Graphics).
- Την υποστήριξη Graph Stylesheets (GSS). [9]

Το συγκεκριμένο εργαλείο υστερεί στο γεγονός ότι δεν υποστηρίζει την OWL γλώσσα αλλά την RDF. Επιπλέον το notation που χρησιμοποιείται στις γραφικές αναπαραστάσεις των υλοποιημένων με το εργαλείο οντολογιών δεν μοιάζει σε κανένα σημείο με την UML αναπαράσταση των μοντέλων, κάτι που κατά κύριο λόγο επιδιώκεται από το GRAMOFONE. Ακόμα το εργαλείο δεν υποστηρίζει την ύπαρξη πολλαπλών

διαγραμμάτων για την υλοποίηση των οντολογιών, αλλά κάθε οντολογία υλοποιείται σε ένα μοναδικό διάγραμμα το οποίο όπως προαναφέρθηκε αποτελεί μειονέκτημα σε σχέση με το GRAMOFONE.

**KAON OI-modeler** (FZI Research Center & AIFB Institute, University of Karlsruhe)
Είναι ένα εργαλείο δημιουργίας και συντήρησης οντολογιών. Το σημαντικότερο χαρακτηριστικό του εργαλείου είναι η υποστήριξη διαχείρισης μεγάλων οντολογιών, μέσω της κλιμακωτής (πολύ-επίπεδης) ανάπτυξής τους. Οι οντολογίες μπορούν να αποθηκευτούν σε έναν εξυπηρετητή (Engineering Server) ο οποίος επιτρέπει μεταξύ άλλων την ταυτόχρονη επεξεργασία των οντολογιών από πολλούς χρήστες. Είναι το μοναδικό από τα εργαλεία που παρουσιάζονται στην παρούσα ενότητα που επιτρέπει την ταυτόχρονη επεξεργασία των οντολογιών από πολλούς χρήστες. Το εργαλείο υποστηρίζει πλήρως Undo και Redo. Υποστηρίζει επίσης την δημιουργία queries μέσω της KAON Query language. Ο λόγος που δεν χρησιμοποιείται η XQuery (XML Query Language) για τη δημιουργία queries αφορά πολιτικές και αποφάσεις που πάρθηκαν από την ομάδα ανάπτυξης του εργαλείου. Η περιήγηση στις οντολογίες είναι δυνατή μέσω διαγραμματικών και δενδρικών components [10].
Το εργαλείο υστερεί στην γλώσσα περιγραφής των μοντέλων που υλοποιούνται και η οποία είναι μία επέκταση της RDF και όχι η OWL. Επιπλέον μειονεκτεί λόγω της αδυναμίας υποστήριξης πολλαπλών διαγραμμάτων, καθώς επιτρέπει την χρήση μόνο ενός διαγράμματος ανά οντολογία.

**MR3 (Meta-Model Management based on RDFs Revision Reflection)** (Shizuoka University & AIST (National Institute of Advanced Industrial Science and Technology))
Το MR3 είναι ένα εργαλείο δημιουργίας και επεξεργασίας RDF και RDFS στοιχείων. Το εργαλείο περιλαμβάνει:

- Ένα γραφικό εργαλείο (editor) επεξεργασίας RDF μοντέλων.
- Ένα γραφικό εργαλείο (editor) επεξεργασίας RDFS μετά-μοντέλων.
- Λειτουργίες διαχείρισης της ορθής αντιστοίχισης (συμφωνίας) μεταξύ των υλοποιημένων μοντέλων και των αντίστοιχων μετα-μοντέλων τους. [11]

Το βασικό μειονέκτημα του εργαλείου είναι η γλώσσα περιγραφής των μοντέλων που υλοποιούνται από αυτό, η οποία είναι η RDF και όχι η OWL.

**OntoTrack: Fast Browsing and Easy Editing of Large Ontologies** (University of Ulm Dept. for Artificial Intelligence, Germany)

Το OntoTrack είναι ένα καινούργιο εργαλείο για το browsing, την δημιουργία και την επεξεργασία οντολογιών "σε μία εικόνα (view)", δηλαδή ο χρήστης έχει την δυνατότητα να ορίσει μία οντολογία χρησιμοποιώντας ένα μοναδικό διάγραμμα. Όπως τονίστηκε και παραπάνω, η ύπαρξη ενός και μόνο διαγράμματος αποτελεί μειονέκτημα σε σύγκριση με το εργαλείο που υλοποιήθηκε στην παρούσα διατριβή. Η γλώσσα περιγραφής των οντολογιών είναι η OWL (επί του παρόντος ένα υποσύνολο της OWL Lite που ονομάζεται OWL Lite-). Συνδυάζει μία σύνθετη γραφική διάταξη και παρέχει λειτουργίες επεξεργασίας οντολογιών, με σκοπό την αποδοτικότερη πλοήγηση και διαχείριση των μεγάλων οντολογιών. Επιπλέον χρησιμοποιεί και έναν εξωτερικό reasoner (RACER reasoner) με σκοπό την παροχή άμεσου feedback των επιπτώσεων των επιλογών μοντελοποίησης που παίρνονται από τον χρήστη κατά την διάρκεια υλοποίησης των οντολογιών. [12]

**RDFAuthor** (Damian Steer)

Το RDFAuthor είναι ένα εργαλείο σχεδιασμένο για τη δημιουργία RDF μοντέλων. Η δημιουργία των διαγραμμάτων βασίζεται σε μεγάλο βαθμό στην μεταφορά δεδομένων μέσα στα διαγράμματα (μέσω Drag And Drop) και της μεταξύ τους συσχέτισης μέσω της χρήσης ενός γραφικού περιβάλλοντος [13].

Το συγκεκριμένο εργαλείο είναι το πιο απλό από τα εργαλεία που παρουσιάζονται στην ενότητα και με το πιο «φτωχό» περιβάλλον εργασίας αλλά και με τις λιγότερες δυνατότητες. Η χρήση της RDF για γλώσσα περιγραφής των μοντέλων που υλοποιούνται από το εργαλείο όπως προαναφέρθηκε δεν καλύπτει τις σύγχρονες ανάγκες (χρειάζεται υποστήριξη της OWL) και η δυνατότητα χρήσης ενός μοναδικού διαγράμματος ανά οντολογία κρίνεται επίσης ανεπαρκής. Επιπλέον η σημειολογία (notation) που χρησιμοποιείται από το εργαλείο δεν μοιάζει με αυτή της UML, γεγονός το οποίο, μεταξύ άλλων, καθιστά το εργαλείο υποδεέστερο του GRAMOFONE.

**Σύγκριση**

Το GRAMOFONE είναι το μόνο εργαλείο που έχει αναπτυχθεί χρησιμοποιώντας την αρχιτεκτονική και τα εργαλεία μοντελοποίησης που παρέχει το πρότυπο MOF του διεθνούς οργανισμού τυποποίησης OMG. Επίσης είναι το μόνο εργαλείο που είναι βασισμένο στη γλώσσα ODM που παρέχει παρόμοια λειτουργικότητα με την OWL-DL αλλά είναι βασισμένη στην αρχιτεκτονική MOF. Επίσης το GRAMOFONE επικοινωνεί και αποθηκεύει τις οντολογίες σε μια βάση δεδομένων που υποστηρίζει την αρχιτεκτονική MOF, παρόλο που σε αυτή τη φάση η επικοινωνία με τη βάση είναι για εξαγωγή ολόκληρης της οντολογίας και όχι επιμέρους τμημάτων της (όπως το KAON). Τέλος είναι το μόνο

εργαλείο που έχει υλοποιηθεί πάνω στην πλατφόρμα Eclipse και αποτελεί plug-in της πλατφόρμας αυτής. Το γεγονός αυτό περιλαμβανόταν στις προδιαγραφές που είχαν τεθεί για την υλοποίηση του GRAMOFONE, όπως είχε οριστεί από τους υπεύθυνους του προγράμματος DBE. Πρέπει επίσης να σημειωθεί πως το GRAMOFONE δεν κάνει χρήση του διαδικτύου, ενώ τα υπόλοιπα εργαλεία χρησιμοποιούν το διαδίκτυο κυρίως για την επαλήθευση των URI's (Uniform Resource Identifier) τα οποία ορίζονται μέσα στις οντολογίες. Στον παρακάτω πίνακα συνοψίζονται τα βασικά χαρακτηριστικά των εργαλείων που μελετήθηκαν και παρουσιάστηκαν. Με κόκκινο χρώμα είναι τονισμένα τα χαρακτηριστικά στα οποία τα υπόλοιπα εργαλεία μειονεκτούν έναντι του GRAMOFONE:

| Εργαλείο | GRAMOFONE | Construct | Protégé + Protégé OWL Plug-in + ezOWL | IsaViz | KAON OI - Modeler | MR3 | OntoTrack | RDFAuthor |
|---|---|---|---|---|---|---|---|---|
| Γλώσσα Περιγραφής | ODM | OWL | OWL | RDF, GSS | KAON extension of RDFS | RDF(S) | OWL Lite in RDFS notation | RDF |
| Υποστήριξη Βάσεων Δεδομένων | XML | Ναι, σχεσιακή | Ναι, σχεσιακή | Όχι | Ναι, σχεσιακή | Όχι | Όχι | Όχι |
| Χρήση – Υποστήριξη WEB | Όχι | URI namespaces | Reference ontologies by URI | URI's, XML namespaces | Ναι | URI's | RDF URI's and namespaces | URI's. (Web links, remote RDF query) |
| Import / Export Formats | ODM / ODM, XML | OWL, XML | RDF, RDFS, DAML+OIL, XML, OWL, Clips, UML, Notation 3, N-Triple | RDF / XML, NOTANIO N 3, N-Triples, SVG, PNG | RDFS, Protégé RDFS | RDF / XML, N-Triple, PNG | RDF / XML, N-Triple | XML, RDF |
| Graph View – Editing | Ναι | Ναι | Ναι | Ναι | Ναι | Ναι | Ναι | Ναι |
| Consistency Checks | Ναι | Ναι | Ναι | Ναι | Ναι | Ναι | Ναι | Ναι |
| UML Notation | Ναι | Ναι | Ναι | Όχι | Ναι | Ναι | Ναι | Όχι |
| License | Μη εμπορικό προϊόν | Εμπορικό προϊόν | Μη εμπορικό προϊόν | Μη εμπορικό προϊόν | Μη εμπορικό προϊόν | Μη εμπορικό προϊόν | Μη εμπορικό προϊόν | Μη εμπορικό προϊόν |
| Multiple Diagrams | Ναι | Ναι | Όχι | Όχι | Όχι | Ναι | Όχι | Όχι |
| Eclipse Plug-in | Ναι | Όχι | Όχι | Όχι | Όχι | Όχι | Όχι | Όχι |
| Multi-user Support | Όχι | Όχι | Όχι | Όχι | Ναι | Όχι | Όχι | Όχι |

**Εικόνα 5: Συγκριτικός πίνακας της παρούσας διατριβής με συσχετιζόμενες εργασίες**

### Ανακεφαλαίωση

Σε αυτό το κεφάλαιο παρουσιάστηκαν οι XML γλώσσες περιγραφής οντολογιών RDF, RDF Schema και OWL της Semantic Web Community, οι οποίες αποτελούν τις πιο γνωστές τεχνολογίες αναπαράστασης της γνώσης σήμερα. Στη συνέχεια παρουσιάστηκε η UML, η οποία αποτελεί την πιο δημοφιλή γλώσσα μοντελοποίησης σήμερα. Τέλος παρουσιάστηκαν εργαλεία που έχουν σχετιζόμενο αντικείμενο μελέτης με το αντικείμενο μελέτης της παρούσας διατριβής, δηλαδή αλληλεπιδραστικά εργαλεία για την ανάπτυξη οντολογιών και παρατέθηκε η μεταξύ τους σύγκριση.

Κεφάλαιο 3

# ΤΕΧΝΟΛΟΓΙΚΗ ΒΑΣΗ

## Εισαγωγή

Το κεφάλαιο ασχολείται με τις τεχνολογίες που μελετήθηκαν και χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής. Τόσο κατά την θεωρητική μελέτη, όσο και κατά το σχεδιασμό και την υλοποίηση της εφαρμογής, μελετήθηκαν και χρησιμοποιήθηκαν ένα πλήθος τεχνολογιών όπως, η πλατφόρμα Eclipse, τα πλαίσια εργασίας Standard Widget Toolkit (SWT), JFace, Graphical Editing Framework (GEF), καθώς επίσης η βάση γνώσης του DBE, το API το οποίο παρέχεται για την πρόσβαση σε αυτή, και άλλες. Για την καλύτερη κατανόηση των θεμάτων υλοποίησης της παρούσης διπλωματικής, ακολουθεί μία μικρή περιγραφή των τεχνολογιών αυτών.

## Η Πλατφόρμα Eclipse

Η πλατφόρμα Eclipse είναι ένα Ενοποιημένο Περιβάλλον Ανάπτυξης Εφαρμογών (Integrated Development Environment / IDE). Πρόκειται για μία πλατφόρμα «γενικής χρήσεως», το οποίο σημαίνει πως η βασική έκδοση της πλατφόρμας δεν παρέχει κάποια ιδιαίτερη - εξειδικευμένη λειτουργικότητα αλλά αποτελεί το θεμέλιο και το μέσο της ανάπτυξης εργαλείων (plug-ins) τα οποία με την προσθήκη τους στην πλατφόρμα επεκτείνουν την λειτουργικότητα που παρέχεται από αυτήν. Η δυνατότητα αυτή της επέκτασης της πλατφόρμας αποτελεί και το σημαντικότερο χαρακτηριστικό της καθώς δίνει την δυνατότητα, με την χρήση των κατάλληλων εργαλείων (plug-ins), της υποστήριξης οποιουδήποτε τύπου δεδομένων (π.χ. Java, HTML, XML κτλ.). Το εργαλείο της παρούσης διπλωματικής (GRAMOFONE) έχει υλοποιηθεί σαν plug-in της πλατφόρμας Eclipse.

Οι τεχνικές προδιαγραφές της πλατφόρμας είναι οι παρακάτω:

i. Υποστήριξη της κατασκευής διαφόρων εργαλείων για την ανάπτυξη εφαρμογών.
ii. Υποστήριξη ενός απεριόριστου συνόλου από προμηθευτές εργαλείων, συμπεριλαμβανομένων και των independent software vendors (ISVs).

iii. Υποστήριξη εργαλείων για την διαχείριση αυθαίρετων τύπων δεδομένων (π.χ. HTML, Java, C, JSP, EJB, XML, και GIF).

iv. Διευκόλυνση της ενσωμάτωσης εργαλείων για την διαχείριση διαφόρων τύπων δεδομένων κατασκευασμένα από διαφορετικούς προμηθευτές εργαλείων.

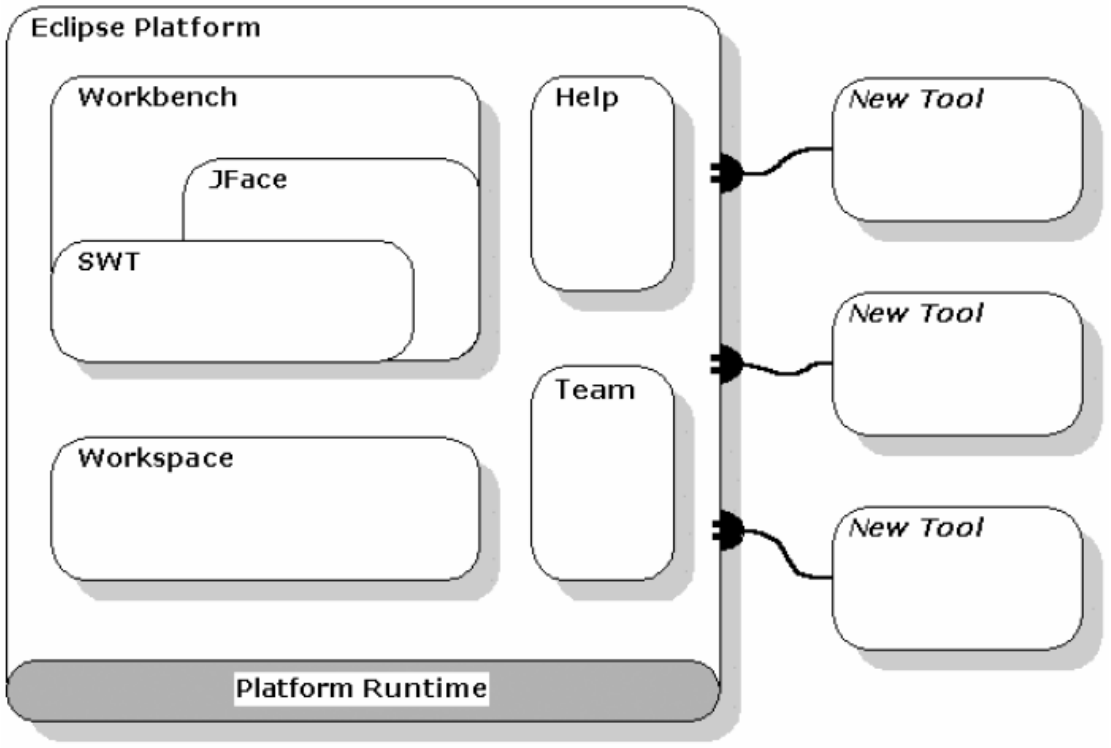
v. Υποστήριξη τόσο γραφικών (GUI-based), όσο και μη γραφικών (non GUI-based) περιβαλλόντων ανάπτυξης εφαρμογών.

vi. Υποστήριξη ενός μεγάλου πεδίου λειτουργικών συστημάτων, συμπεριλαμβανομένων των Windows® και του Linux™.

vii. Η εκμετάλλευση της δημοτικότητας της γλώσσας προγραμματισμού Java για την ανάπτυξη εργαλείων.

**Eclipse Platform Αρχιτεκτονική**

Τα σημαντικότερα δομικά μέρη (components) της πλατφόρμας Eclipse φαίνονται στην εικόνα 6. [17]




**Εικόνα 6: Eclipse Platform Architecture**

Παρατηρούμε από την εικόνα 6 πως η αρχιτεκτονική της πλατφόρμας είναι προσανατολισμένη στη χρήση επιπρόσθετων εργαλείων (plug-ins) πάνω σε έναν βασικό πυρήνα λειτουργικότητας. Η πλατφόρμα Eclipse αποτελεί την βάση πάνω στην οποία προστίθονται νέα εργαλεία (plug-ins) με σκοπό την επέκταση της λειτουργικότητάς της. Όλα τα δομικά μέρη (components) της πλατφόρμας είναι απαραίτητα για την γραφική έκδοση (GUI-based) της πλατφόρμας. Η μη γραφική έκδοση (Text-based) της πλατφόρμας

δεν θα χρειαζόταν το δομικό στοιχείο "Workbench" το οποίο παρέχει τα γραφικά χαρακτηριστικά της πλατφόρμας. Παρακάτω ακολουθεί η περιγραφή των δομικών μερών της αρχιτεκτονικής της πλατφόρμας Eclipse.

**Ο Πυρήνας της Πλατφόρμας Eclipse και ο Ρόλος των Plug-ins**

Εκτός από ένα μικρό πυρήνα γνωστό ως Platform Runtime, όλη η λειτουργικότητα της πλατφόρμας βρίσκεται στα plug-ins. Ένα plug-in είναι η μικρότερη λειτουργική μονάδα της πλατφόρμας Eclipse η οποία μπορεί να αναπτυχθεί και να παραδοθεί ξεχωριστά. Συνήθως ένα εργαλείο δημιουργείται σαν ένα μοναδικό plug-in, ενώ υπάρχουν περιπτώσεις εργαλείων που διαχωρίζουν την λειτουργικότητά τους σε αρκετά διαφορετικά plug-ins. Το GRAMOFONE αποτελεί και αυτό ένα plug-in. Η γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάπτυξη των plug-ins είναι η Java. Ένα τυπικό plug-in αποτελείται από μία JAR (Java ARchive) βιβλιοθήκη που περιέχει Java κώδικα, και άλλους πόρους όπως εικόνες, πρότυπα web (web templates), κατάλογοι μηνυμάτων (message catalogs), βιβλιοθήκες γηγενούς κώδικα (native code libraries) κτλ. Κάθε plug-in αποτελείται από ένα αρχείο-μανιφέστο (manifest file) που δηλώνει τις αλληλοσυνδέσεις του συγκεκριμένου plug-in με άλλα plug-ins. Το μοντέλο αλληλοσύνδεσης είναι απλό: ένα plug-in δηλώνει έναν οποιοδήποτε αριθμό από σημεία επέκτασης (extension points), και έναν οποιοδήποτε αριθμό από επεκτάσεις (extensions) σε ένα ή περισσότερα σημεία επέκτασης άλλων plug-ins. Τα σημεία επέκτασης ενός plug-in μπορούν να χρησιμοποιηθούν (επεκταθούν) από άλλα plug-ins με σκοπό την επέκταση της λειτουργικότητας του αρχικού plug-in. Τα plug-ins που επεκτείνουν την λειτουργικότητα χρησιμοποιούν δηλώσεις που ονομάζονται επεκτάσεις (extensions). Για παράδειγμα, το plug-in που είναι υπεύθυνο για το δομικό στοιχείο "workbench" της αρχιτεκτονικής της πλατφόρμας Eclipse που είδαμε στην εικόνα 6 ορίζει ένα σημείο επέκτασης για την δήλωση των προτιμήσεων των χρηστών (user preferences). Ένα plug-in μπορεί να ορίσει τις δικές του προτιμήσεις χρηστών ορίζοντας επεκτάσεις σε αυτό το σημείο επέκτασης. Ένα παράδειγμα αλληλοσύνδεσης δύο plug-ins στο πλαίσιο του προγράμματος DBE αποτελούν ο BML Editor, ο οποίος χρησιμοποιείται για τη δημιουργία μοντέλων περιγραφής εταιρειών, και το εργαλείο Semantic Discovery Tool, το οποίο χρησιμοποιείται για την ανεύρεση μοντέλων περιγραφής εταιρειών τα οποία έχουν ήδη δημιουργηθεί. Πιο συγκεκριμένα, ο BML Editor ενσωματώνει στη λειτουργικότητά του το Semantic Discovery Tool. Μέσω της ενσωμάτωσης των plug-ins στην πλατφόρμα Eclipse δίνεται η δυνατότητα της ευκολότερης και αποδοτικότερης διαχείρισης της αλληλεπίδρασης μεταξύ των plug-ins αυτών.

Το GRAMOFONE αποτελείται από ένα αρχείο-μανιφέστο (plugin.xml), μία JAR βιβλιοθήκη (ontologyEditor.jar), οκτώ βοηθητικές βιβλιοθήκες για την σύνδεση και την επικοινωνία με τη Βάση Γνώσης (Knowledge Base) και τέλος τα εικονίδια που χρησιμοποιούνται στο περιβάλλον εργασίας του GRAMOFONE.

Κατά την εκκίνηση της πλατφόρμας το Platform Runtime ανακαλύπτει το σύνολο των διαθέσιμων plug-ins, διαβάζει τα αρχεία-μανιφέστα του καθενός, και δημιουργεί στην κύρια μνήμη έναν κατάλογο (registry) για αυτά. Μετά την εκκίνηση της πλατφόρμας δεν μπορούν να προστεθούν άλλα plug-ins. Η ενεργοποίηση ενός plug-in γίνεται όταν χρειαστεί να τρέξει ο κώδικάς του, και όχι υποχρεωτικά κατά την εκκίνηση. Αυτό έχει σαν συνέπεια ο χρόνος εκκίνησης να είναι αρκετά πιο σύντομος.

**Χώροι Εργασίας (Workspaces)**

Τα διάφορα εργαλεία (plug-ins) που λειτουργούν στην πλατφόρμα Eclipse ενεργούν πάνω σε αρχεία που βρίσκονται στο χώρο εργασίας (workspace) του χρήστη. Ένας χώρος εργασίας αποτελείται από ένα ή περισσότερα ανώτατα έργα (top-level projects), όπου κάθε έργο (project) αντιστοιχεί σε ένα υποκατάλογο (folder) στο τοπικό σύστημα αρχείων ο οποίος έχει όνομα ορισμένο από το χρήστη (user-defined). Οι όροι που χρησιμοποιούνται στους χώρους εργασίας (workspaces) είναι πόροι όπως έργα (projects), αρχεία και υποκατάλογοι, μηχανισμοί καταγραφής αλλαγών (history mechanisms), σημειώματα (markers) όπως bookmarks, debugger breakpoints, to-do list items κτλ.

**Workbench και UI Toolkits (SWT και JFace)**

Το περιβάλλον εργασίας της πλατφόρμας Eclipse έχει δημιουργηθεί γύρω από έναν πάγκο εργασίας (workbench) ο οποίος παρέχει την γενική δομή και παρουσιάζει ένα επεκτάσιμο περιβάλλον στο χρήστη. Ο πάγκος εργασίας (workbench) έχει δημιουργηθεί μέσω της χρήσης δύο εργαλείων, του Standard Widget Toolkit (SWT) και του JFace.

Το SWT παρέχει ένα κοινό προς τους χρήστες και ανεξάρτητο από το λειτουργικό σύστημα API για widgets και γραφικά δομικά στοιχεία (components) υλοποιημένα με τέτοιο τρόπο ώστε να επιτρέπεται η στενή ενσωμάτωσή τους με το τοπικό λειτουργικό σύστημα. Η στενή ενσωμάτωση με το τοπικό λειτουργικό σύστημα πάνω στο οποίο λειτουργεί η πλατφόρμα Eclipse δεν είναι μόνο ζήτημα αισθητικής (π.χ. χρήση του γηγενούς συστήματος γραφικών – look and feel). Το SWT αλληλεπιδρά με λειτουργίες του τοπικού συστήματος όπως είναι το Drag And Drop (DND), και μπορεί να χρησιμοποιήσει δομικά στοιχεία (components) που έχουν αναπτυχθεί στο τοπικό σύστημα όπως είναι τα Windows ActiveX controls. Ολόκληρο το περιβάλλον εργασίας της πλατφόρμας Eclipse και των εργαλείων που

ενσωματώνονται σε αυτήν χρησιμοποιούν το SWT για την παρουσίαση πληροφοριών στον τελικό χρήστη.

Το πλαίσιο εργασίας JFace είναι ένα εργαλείο για την υποστήριξη των διεπαφών χρήστη (User Interface) που αποτελείται από κλάσεις για την διεκπεραίωση κοινών προγραμματιστικών λειτουργιών σε σχέση με αυτό. Το JFace είναι ανεξάρτητο από το τοπικό λειτουργικό σύστημα τόσο στο API που ορίζει όσο και στην υλοποίησή του, και είναι σχεδιασμένο να συνεργάζεται με το SWT χωρίς να το καλύπτει. Περιλαμβάνει συνηθισμένα στοιχεία διεπαφών όπως είναι τα μητρώα εικόνων και γραμματοσειρών (image and font registries), τα παράθυρα διαλόγου (dialogs), οι προτιμήσεις εμφάνισης (preferences), τα wizard frameworks, και οι αναφορές προόδου για μακροχρόνιες λειτουργίες.

Τα πιο ενδιαφέρονται χαρακτηριστικά του JFace είναι οι μηχανισμοί που παρέχει για τον καθορισμό ενεργειών (actions) και μηχανισμών προβολής (viewers). Ο μηχανισμός ενεργειών επιτρέπει το ορισμό εντολών από το χρήστη ανεξάρτητα από την τοποθεσία στην οποία αυτή έχει ισχύ μέσα στο περιβάλλον εργασίας. Μία ενέργεια (action) αναπαριστά μία εντολή η οποία μπορεί να ενεργοποιηθεί από το χρήστη μέσω ενός κουμπιού (button), μιας επιλογής κάποιου μενού (menu item), ή ενός στοιχείου σε κάποια μπάρα εργαλείων (tool bar).

Οι μηχανισμοί προβολής (viewers) αποτελούν προσαρμογείς (adapters), βασισμένους σε μοντέλα (model-based), κάποιων συγκεκριμένων SWT widgets. Οι μηχανισμοί αυτοί διεκπεραιώνουν την κοινή (γνώριμη) συμπεριφορά των στοιχείων και παρέχουν υψηλότερου επιπέδου έννοιες από αυτές που είναι διαθέσιμες από τα SWT widgets. Παραδείγματα τέτοιων μηχανισμών είναι οι μηχανισμοί προβολής λιστών (list viewers), οι μηχανισμοί προβολής δένδρων (tree viewers), οι μηχανισμοί προβολής πινάκων (table viewers) κτλ.

Η επιφάνεια εργασίας (workbench), σε αντίθεση με τα πλαίσια εργασίας SWT και JFace τα οποία είναι γενικής χρήσης εργαλεία διεπαφών, διαμορφώνει την προσωπικότητα του περιβάλλοντος εργασίας της πλατφόρμας Eclipse, και παρέχει τις δομές με τις οποίες τα εργαλεία (plug-ins) αλληλεπιδρούν με τον χρήστη. Για τον λόγο αυτό το workbench έχει ταυτιστεί σαν έννοια με το περιβάλλον εργασίας της πλατφόρμας Eclipse. Από πλευράς χρήστη ένα παράθυρο του workbench αποτελείται οπτικά από περιοχές προβολής (views) και περιοχές επεξεργασίας (editors). Με τον όρο perspective (όψη) εννοείται μία προκαθορισμένη διάταξη από τέτοιες περιοχές οι οποίες είναι τοποθετημένες σε προκαθορισμένες θέσεις. Στην πλατφόρμα Eclipse μπορούν να οριστούν πολλά διαφορετικά perspectives και είναι δυνατό την ίδια χρονική στιγμή να είναι ανοιγμένα περισσότερα του ενός. Ωστόσο μόνο ένα perspective μπορεί να είναι ενεργό σε κάθε χρονική στιγμή.

Το GRAMOFONE για παράδειγμα ορίζει το δικό του perspective το οποίο ονομάζεται Ontology Perspective. Στο perspective αυτό ορίζονται περιοχές προβολής (views) και επεξεργασίας (editors) που χρησιμοποιούνται από το εργαλείο καθώς και οι θέσεις τους στο περιβάλλον εργασίας, αποκρύπτοντας την ίδια στιγμή περιοχές προβολής (views) ή επεξεργασίας (editors) που θα ήταν άχρηστες για το εργαλείο. Οι περιοχές προβολής (views) και οι περιοχές επεξεργασίας (editors) έχουν τη μορφή παραθύρων (windows).
Οι περιοχές επεξεργασίας (editors) επιτρέπουν στο χρήστη το άνοιγμα, την επεξεργασία και το σώσιμο αντικειμένων. Όταν μια τέτοια περιοχή είναι ενεργή, μπορεί να συνεισφέρει διάφορες επιλογές ενεργειών (actions) στα μενού (menus) και στη μπάρα εργαλείων (tool bar) της επιφάνειας εργασίας. Οι περιοχές επισκόπησης (views) παρέχουν πληροφορίες για το αντικείμενο με το οποίο ασχολείται ο χρήστης στην επιφάνεια εργασίας και παρέχουν στον χρήστη τη δυνατότητα τροποποίησης του αντικειμένου αυτού. Τροποποιήσεις που γίνονται σε μία περιοχή επισκόπησης (όπως αλλάζοντας την τιμή ενός χαρακτηριστικού του υπό-επεξεργασία αντικειμένου) γενικά σώζονται άμεσα, και οι αλλαγές αντανακλώνται αμέσως στα τμήματα του περιβάλλοντος εργασίας τα οποία επηρεάζονται από την αλλαγή αυτή.

**Υποστήριξη Ομαδικής Εργασίας (Team Support)**

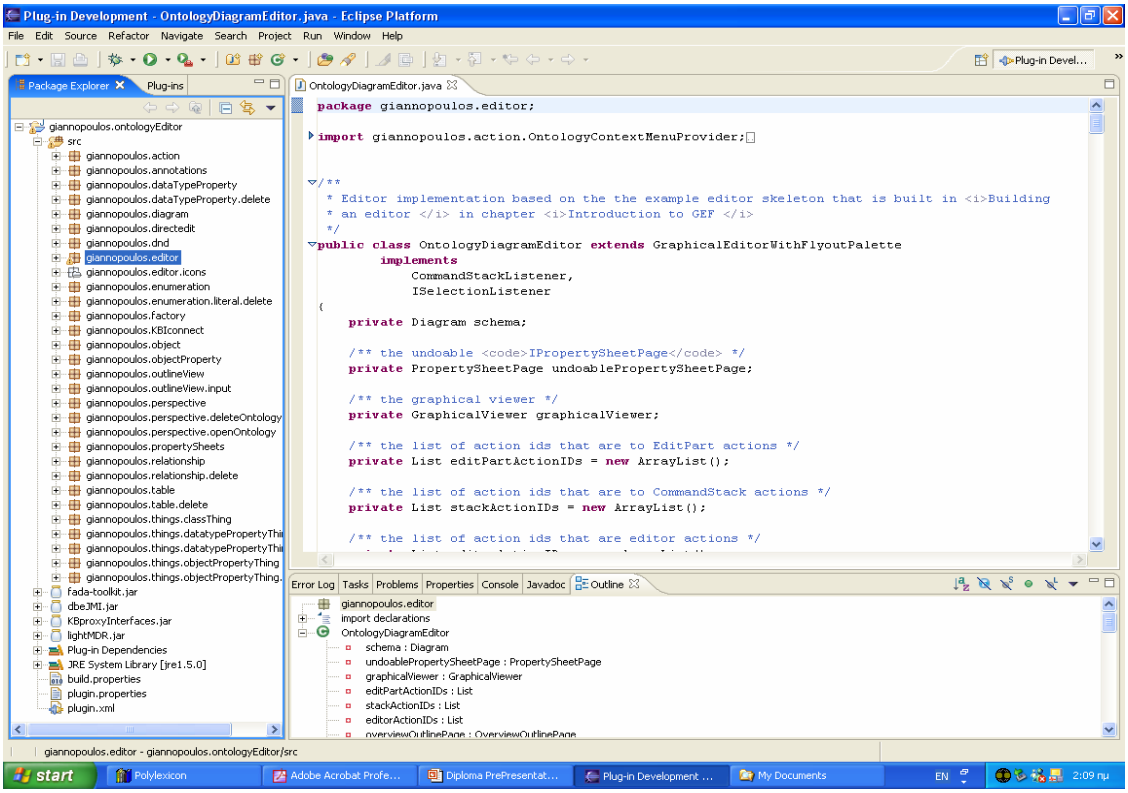
Η πλατφόρμα Eclipse επιτρέπει σε έργα (projects) που βρίσκονται μέσα στο χώρο εργασίας (workspace) να έχουν συγκεκριμένες εκδόσεις (versions) και παρέχει τη δυνατότητα διαχείρισης των ιδιοτήτων τους μέσω της χρήσης κατάλληλων μηχανισμών αποθήκευσης διαμοιραζόμενων πόρων (team repositories). Η πλατφόρμα διαθέτει σημεία επέκτασης και ένα API ορισμού repository, τα οποία επιτρέπουν τον ορισμό νέων team repositories. Γενικά, είναι δυνατή η συνύπαρξη πολλαπλών τέτοιων μηχανισμών (από διαφορετικούς προμηθευτές πιθανόν) στην πλατφόρμα Eclipse. Επίσης, η πλατφόρμα περιλαμβάνει την υποστήριξη μηχανισμών ομαδικής ανάπτυξης κώδικα (Concurrent Versions System / CVS) που είναι προσβάσιμοι μέσω πρωτοκόλλων είτε pserver είτε ssh. Με τη χρήση αυτών των μηχανισμών ένα έργο (project) μπορεί να τροποποιηθεί ή να σχολιαστεί από τα μέλη της ομάδας του συγκεκριμένου αποθηκευτικού χώρου τα οποία μπορεί να βρίσκονται ακόμα και σε διαφορετικές χώρες.

**Μηχανισμός Βοήθειας (Help)**

Ο μηχανισμός βοήθειας της πλατφόρμας Eclipse επιτρέπει στα εργαλεία να ορίζουν και να συνεισφέρουν τεκμηρίωση (documentation) σε ένα ή περισσότερα online εγχειρίδια. Για παράδειγμα, ένα εργαλείο συνήθως συνεισφέρει τεκμηρίωση με την μορφή ενός οδηγού για

το χρήστη και τεκμηρίωση των διεπαφών προγραμματισμού (API) (αν περιλαμβάνει κάτι τέτοιο) σε ένα ξεχωριστό οδηγό για τον προγραμματιστή.

Μία εικόνα του περιβάλλοντος ανάπτυξης plug-ins (Plug-in Development Environment / PDE) της πλατφόρμας φαίνεται στην εικόνα 7. Το PDE αποτελεί το ενυπάρχων (built-in) perspective της πλατφόρμας που χρησιμοποιείται για την υλοποίηση plug-ins και συνεπώς και του GRAMOFONE.




**Εικόνα 7: Plug-in Development (PDE) Screenshot**

## Επίλογος

Η πλατφόρμα Eclipse παρέχει έναν πυρήνα από δομικά στοιχεία γενικού περιεχομένου και ένα σύνολο διεπαφών προγραμματισμού (APIs) όπως είναι ο χώρος εργασίας (workspace) και η επιφάνεια εργασίας (workbench), και διάφορα σημεία επέκτασης μέσω των οποίων είναι δυνατή η ενσωμάτωση νέων λειτουργιών. Μέσω αυτών των σημείων επέκτασης, εργαλεία υλοποιημένα σαν ξεχωριστά επιπρόσθετα (plug-ins) μπορούν να επεκτείνουν τη βασική πλατφόρμα Eclipse. Στον τελικό χρήστη παρουσιάζεται ένα περιβάλλον ανάπτυξης εφαρμογών (IDE) εξειδικευμένο με βάση το σύνολο των διαθέσιμων επιπρόσθετων (plug-in) εργαλείων. Ωστόσο, οι δυνατότητες επέκτασης δεν σταματούν εδώ. Τα εργαλεία μπορούν να ορίσουν νέα δικά τους σημεία επέκτασης και δικές τους διεπαφές προγραμματισμού (APIs) και με τον τρόπο αυτό να αποκτούν τη μορφή δομικών υλικών και να αποτελούν σημεία ενσωμάτωσης άλλων εργαλείων.

**Graphical Editing Framework (GEF)**

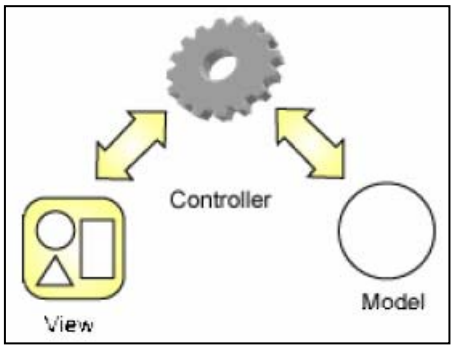
Τα γραφικά μέρη του GRAMOFONE δημιουργήθηκαν με τη βοήθεια του πλαισίου εργασίας GEF (Graphical Editing Framework). Το GEF επιτρέπει στους δημιουργούς εφαρμογών να δημιουργούν έναν πλούσιο γραφικό επεξεργαστή (graphical editor) βασισμένο στο μοντέλο της εφαρμογής αυτής καθαυτής. Το πλαίσιο εργασίας GEF αποτελείται από 2 επιπρόσθετα (plug-ins), το org.eclipse.draw2d και το org.eclipse.gef. Τα επιπρόσθετα αυτά παρέχουν όλη τη λειτουργικότητα που περιλαμβάνει το πλαίσιο εργασίας GEF. Ο δημιουργός μέσω της χρήσης των βιβλιοθηκών που παρέχονται από τα 2 προαναφερθέντα επιπρόσθετα (plug-ins) μπορεί να εκμεταλλευτεί πολλές κοινές λειτουργίες που παρέχονται στο GEF και / ή να τις επεκτείνει για ένα συγκεκριμένο τομέα εφαρμογών (application domain). [19]

Το GEF υποθέτει πως υπάρχει ένα μοντέλο για το οποίο ο δημιουργός της εφαρμογής επιθυμεί να κάνει δυνατή την αναπαράστασή του και την επεξεργασία του με γραφικό τρόπο. Για το σκοπό αυτό το GEF παρέχει μηχανισμούς επισκόπησης (viewers) οι οποίοι μπορούν να χρησιμοποιηθούν οπουδήποτε μέσα στην επιφάνεια εργασίας (workbench) της πλατφόρμας Eclipse. Όπως και στην περίπτωση του πλαισίου εργασίας JFace έτσι και στην περίπτωση του πλαισίου εργασίας GEF οι μηχανισμοί επισκόπησης (viewers) αποτελούν προσαρμογείς (adapters) κάποιων συγκεκριμένων SWT γραφικών δομών (widgets). Αλλά η ομοιότητά τους φτάνει ως εδώ. Οι μηχανισμοί επισκόπησης του GEF βασίζονται στην αρχιτεκτονική Μοντέλου – Προβολής – Ελεγκτών (Model-View-Controller / MVC αρχιτεκτονική) η οποία και θα εξηγηθεί στη συνέχεια. [18]

Η κεντρική ιδέα της MVC αρχιτεκτονικής είναι πως κάθε στοιχείο του Μοντέλου μίας εφαρμογής (Model), το οποίο ο χρήστης επιθυμεί να έχει τη δυνατότητα να τροποποιεί με γραφικό τρόπο, μπορεί να αναπαρασταθεί με μία γραφική αναπαράσταση (View). Την επικοινωνία και την σύνδεση μεταξύ των δύο δομικών στοιχείων (Model και View) αναλαμβάνουν οι Ελεγκτές (Controllers), οι οποίοι φροντίζουν να είναι διαρκώς συγχρονισμένα τα στοιχεία του Μοντέλου με τις αντίστοιχες γραφικές τους αναπαραστάσεις. Οι ελεγκτές (controllers) στην MVC αρχιτεκτονική γεφυρώνουν το μοντέλο με την γραφική του αναπαράσταση (εικόνα 8). Κάθε ελεγκτής (controller, ή EditPart όπως ονομάζονται), είναι υπεύθυνος για την αντιστοίχιση (mapping) του μοντέλου με την γραφική του αναπαράσταση καθώς και για την πραγματοποίηση αλλαγών στο μοντέλο. Ο ελεγκτής επίσης «παρακολουθεί» το μοντέλο, και ενημερώνει τη γραφική του αναπαράσταση ώστε να αντανακλά τις αλλαγές που πραγματοποιούνται στην κατάσταση του μοντέλου. Τα αντικείμενα με τα οποία ο χρήστης αλληλεπιδρά είναι οι ελεγκτές.

Ας θεωρήσουμε για παράδειγμα την έννοια «Class» η οποία ορίζεται στο μετά-μοντέλο ODM και την οποία επιθυμούμε να είμαστε σε θέση τόσο να την αναπαραστήσουμε όσο και να την τροποποιήσουμε γραφικά. Για το σκοπό αυτό πραγματοποιούμε τις ακόλουθες ενέργειες. Σε πρώτο στάδιο υλοποιούμε την έννοια με τη χρήση Java κλάσεων. Το «Model» τμήμα της εικόνας 8 περιλαμβάνει την υλοποίηση αυτή (Java κλάσεις) της έννοιας. Επιλέγουμε για γραφική αναπαράσταση της έννοιας ένα ορθογώνιο το οποίο θα περιλαμβάνει στην κορυφή του το όνομα της κλάσης. Η γραφική αυτή αναπαράσταση ονομάζεται «figure», υλοποιείται σε κατάλληλη Java κλάση, και περιλαμβάνεται στο τμήμα «View» της εικόνας 8. Για τη σύνδεση της έννοιας «Class» με τη γραφική της αναπαράσταση ορίζεται ένας «Controller» (ή EditPart) ο οποίος αναλαμβάνει την αντιστοίχιση της έννοιας «Class» με τη γραφική της αναπαράσταση και επιπλέον ορίζει πολιτικές (policies) για τη διαχείριση ενεργειών που πραγματοποιούνται γραφικά, ενώ παράλληλα λειτουργεί ως «listener» των ενεργειών που πραγματοποιεί ο χρήστης γραφικά. Έτσι εάν οριστεί πως είναι επιθυμητή η δυνατότητα της άμεσης γραφικής επεξεργασίας του ονόματος της έννοιας (μέσω του ορισμού μίας Direct Edit Policy), τότε μέσω κατάλληλης γραφικής επιλογής από το χρήστη (μονό αριστερό κλικ πάνω στο όνομα της γραφικής αναπαράστασης) ενεργοποιείται η λειτουργία και ο «Controller» αναλαμβάνει την διεκπεραίωσή της (ενημέρωση τόσο του μοντέλου όσο και της γραφικής αναπαράστασης για την αλλαγή). Ο «Controller» υλοποιείται και αυτός σε μία Java κλάση.

Η αναπαράσταση των μοντέλων των εφαρμογών με τη χρήση του GEF γίνεται μέσω μηχανισμών που ονομάζονται "viewers". Το GEF παρέχει δύο τύπους μηχανισμών επισκόπησης (viewers): τους γραφικούς και αυτούς που έχουν δενδρική μορφή. Ο καθένας παρέχει ένα διαφορετικό τύπο γραφικής αναπαράστασης (view). Ο γραφικός viewer χρησιμοποιεί σχήματα (figures) τα οποία εμφανίζονται (ζωγραφίζονται) πάνω στον καμβά του SWT. Τα σχήματα (figures) ορίζονται με τη χρήση του Draw2D plug-in, το οποίο περιλαμβάνεται στο GEF. Ο μηχανισμός δενδροειδούς προβολής (TreeViewer) χρησιμοποιεί μία δομή δένδρου που παρέχεται από το SWT καθώς και άλλα γραφικά στοιχεία (TreeItems) για την γραφική του αναπαράσταση.




**Εικόνα 8: Model-View-Controller**

Το GEF είναι εντελώς ανεξάρτητο από εφαρμογές και παρέχει βασικές υποδομές για τη δημιουργία σχεδόν οποιασδήποτε εφαρμογής, συμπεριλαμβανομένων των: επεξεργαστών διαγραμμάτων δραστηριοτήτων (activity diagram editors), εφαρμογές κατασκευής γραφικών διεπαφών (GUI builders), επεξεργαστές διαγραμμάτων κλάσεων (class diagram editors), επεξεργαστές μηχανών καταστάσεων (state machines editors), ακόμα και επεξεργαστές τύπου WYSIWYG (What You See Is What You Get).

Τα χαρακτηριστικά του org.eclipse.draw2d plug-in και κατά συνέπεια και του GEF (του οποίου το plug-in αποτελεί τμήμα) είναι τα ακόλουθα:

- Υποστήριξη της έγκυρης αναπαράστασης των γραφικών τμημάτων της εφαρμογής και της ομαλής γραφικής τους διάταξής. Ουσιαστικά το χαρακτηριστικό αναφέρεται στην ικανότητα του plug-in που αφορά την υψηλή ποιότητα των γραφικών στοιχείων που ορίζονται μέσα από αυτό.
- Ποικίλες υλοποιήσεις των γραφικών αναπαραστάσεων (figures) και των διατάξεών (layout) τους. Τα σχήματα (figures) που ορίζονται με τη χρήση του plug-in μπορούν να έχουν οποιαδήποτε μορφή και σχήμα (π.χ. ορθογώνια, τετράγωνα κτλ.).
- Υποστήριξη ορίων (borders) στα σχήματα (figures).
- Υποστήριξη δεικτών ποντικιού (cursors) και επεξηγηματικών μηνυμάτων (tooltips).
- Υποστήριξη γραφικών συνδέσεων (connections) με τα εξής χαρακτηριστικά:
    - Δυνατότητα προσάρτησης της σύνδεσης (των δύο άκρων της σύνδεσης) σε καθορισμένα από το χρήστη σημεία τα οποία βρίσκονται πάνω στα γραφικά στοιχεία που συνδέονται με τη σύνδεση (Anchoring).
    - Δυνατότητα ορισμού της διαδρομής που ακολουθεί μία σύνδεση όταν συνδέει δύο γραφικά στοιχεία (Routing).
    - Δυνατότητα διακόσμησης των συνδέσεων.
- Πολλαπλά, διαφανή στρώματα (layers).
- Ευμετάβλητα συστήματα συντεταγμένων.

- Δυνατότητα ορισμού μίας περιοχής επισκόπησης (view) η οποία παρουσιάζει τα διαγράμματα που δημιουργούνται από την εκάστοτε εφαρμογή σε κλίμακα (σμίκρυνση).
- Δυνατότητα εκτύπωσης των διαγραμμάτων που δημιουργούνται από την εκάστοτε εφαρμογή.

Τα χαρακτηριστικά του org.eclipse.gef plug-in και κατά συνέπεια και του GEF (του οποίου το plug-in αποτελεί τμήμα) είναι τα ακόλουθα:

- Εργαλεία που παρέχουν τις δυνατότητες επιλογής (selection) και δημιουργίας (creation) γραφικών στοιχείων. Επιπλέον εργαλεία που παρέχουν τη δυνατότητα δημιουργίας συνδέσεων (connections) και εργαλεία για την ταυτόχρονη επιλογή δύο ή περισσοτέρων γραφικών στοιχείων (marquee).
- Μία παλέτα για την επίδειξη των παραπάνω εργαλείων.
- Λαβές (handles) για την αλλαγή του μεγέθους (resizing) των γραφικών αντικειμένων και για την κάμψη (bending) των συνδέσεων (connections).
- Δύο είδη μηχανισμών επισκόπησης όπως περιγράφηκαν πιο πάνω (Γραφικοί και Δενδρικοί).
- Ένα πλαίσιο ελεγκτών (controllers) για την αντιστοίχιση (mapping) του μοντέλου της εφαρμογής με την γραφική του αναπαράσταση. Το πλαίσιο αυτό περιλαμβάνει:
    - Ορισμός πολιτικών των plug-ins (plug-in policies) με σκοπό την αντιστοίχιση την αλληλεπιδράσεων με την γραφική αναπαράσταση του μοντέλου στο ίδιο το μοντέλο.
    - Ποικίλες υλοποιήσεις για την επίδειξη ανάδρασης (feedback) και την προσθήκη λαβών (handles) επιλογής (selection) γραφικών στοιχείων.
    - Ποικίλοι τύποι αιτημάτων (requests) και εργαλεία ή ενέργειες (actions) που στέλνουν αυτά τα αιτήματα στους ελεγκτές (controllers).
- Υποστήριξη των λειτουργιών Ακύρωσης / Επανάληψης (Undo/Redo).

**DBE Knowledge Base (KB)**

Η βάση γνώσης (Knowledge Base, από εδώ και στο εξής KB) του DBE έχει σαν σκοπό την παροχή μίας κοινής και συνεπούς περιγραφής του DBE κόσμου και της δυναμικής του, καθώς επίσης και των εξωτερικών παραγόντων της βιόσφαιρας που τον επηρεάζουν. [16]

Το περιεχόμενο της KB περιλαμβάνει μεταξύ άλλων και αναπαραστάσεις οντολογιών. Η λειτουργικότητα της KB είναι πολυδιάστατη. Ωστόσο η KB θα περιγραφεί με γνώμονα την χρησιμότητά της για το εργαλείο που υλοποιήθηκε στο πλαίσιο της παρούσας διατριβής (GRAMOFONE). Τα κύρια χαρακτηριστικά της υλοποίησης της KB είναι τα ακόλουθα:

- Η ΚΒ είναι βασισμένη σε πρότυπα (Standards-Based): Η ΚΒ υποστηρίζει τα πρότυπα που έχουν θεσπιστεί από τον οργανισμό OMG για την μοντελοποίηση και την ανταλλαγή μοντέλων (πρότυπα για την οργάνωση της πληροφορίας και την διακίνησή της) (MOF 1.4, και XMI 1.2). Η υιοθέτηση των προϋπαρχόντων προτύπων, εγγυάται την συμβατότητα με τις βιομηχανικές κατευθύνσεις, και καθιστά δυνατό τον διαμοιρασμό των μοντέλων μεταξύ των διαφόρων εργαλείων μοντελοποίησης.
- Η ΚΒ είναι επεκτάσιμη (Extensible): Η ΚΒ επεκτείνεται εύκολα ώστε να υποστηρίζει καινούργια μοντέλα πληροφορίας και πηγές πληροφοριών (που δεν υποστηρίζει ήδη). Δεν υπάρχει όριο στις πληροφοριακές πηγές που μπορούν να προστεθούν ή να μοντελοποιηθούν.
- Η ΚΒ υποστηρίζει πολλαπλά μετά-μοντέλα και μοντέλα: Η ΚΒ περιλαμβάνει πολλαπλά μετά-μοντέλα περιγραφής Οντολογιών. Συμπληρωματικά μετά-μοντέλα, τα οποία πρόκειται να χρησιμοποιηθούν μελλοντικά, μπορούν εύκολα να υλοποιηθούν, να υποστηριχθούν και να διαχειριστούν. Η ΚΒ παρέχει τη δυνατότητα στους επαγγελματίες χρήστες να μοντελοποιούν τις επιχειρήσεις και τις υπηρεσίες που προσφέρουν, καθώς επίσης να χρησιμοποιούν, να ενσωματώνουν ή να τροποποιούν υπάρχοντα μοντέλα.
- Η ΚΒ υποστηρίζει πολλαπλά διαφορετικά επίπεδα παρουσίασης της πληροφορίας (Information Views): Η ΚΒ υποστηρίζει διαφορετικές πληροφοριακές απόψεις. Με άλλα λόγια, υποστηρίζει διαφορετικά επίπεδα πρόσβασης στα ίδια μεταδεδομένα από διαφορετικές οπτικές πλευρές (perspectives).
- Η ΚΒ είναι ανεξάρτητη της πλατφόρμας που την χρησιμοποιεί (Platform Independent): Η ΚΒ είναι σχεδιασμένη και υλοποιημένη αποκλειστικά για την Java πλατφόρμα, και κατά συνέπεια είναι ανεξάρτητη από την πλατφόρμα που την χρησιμοποιεί και μπορεί να εγκατασταθεί από τις επιχειρήσεις ανεξάρτητα των λειτουργικών συστημάτων που χρησιμοποιούν. Είναι συμβατή με τα πρότυπα J2EE (Java 2 Platform, Enterprise Edition), JMI (Java Metadata Interface), JDBC (Java Database Connectivity) κτλ. που έχουν υιοθετηθεί από την Java κοινότητα (Java Community Process) και με τα υπάρχοντα XML πρότυπα.

Η υποδομή της ΚΒ ακολουθεί την MOF προσέγγιση μετά-δεδομένων θεσπισμένη από τον οργανισμό OMG. Η ΚΒ είναι σε θέση να διαχειρίζεται όλους τους τύπους πληροφοριών μέσα στο DBE (μεταδεδομένα και δεδομένα) και να εξάγει χρήσιμες πληροφορίες. Η ΚΒ διαχειρίζεται MOF μετά-μοντέλα, και στιγμιότυπα των μετά-μοντέλων αυτών (μοντέλα)

παρέχοντας την πλήρη λειτουργικότητα ενός MOF repository (αποθήκη). Επιπλέον χρησιμοποιεί XMI έγγραφα για την ανταλλαγή μετά-δεδομένων και δεδομένων.

Από τεχνικής άποψης, η αρχιτεκτονική της KB βασίζεται στον συνδυασμό ενός repository συμβατού με το MOF πρότυπο και ενός συστήματος διαχείρισης δεδομένων. Η KB ενσωματώνει το Netbeans Metadata repository (MDR) και χρησιμοποιεί σαν μέσο αποθήκευσης μία java σχεσιακή βάση δεδομένων (Apache Derby Java Database).

Το Ontology Definition Metamodel (ODM) μετά-μοντέλο έχει εισαχθεί στην KB. Η KB παρέχει βασικές λειτουργίες για την επεξεργασία, την αποθήκευση και την ανάκτηση μοντέλων που ακολουθούν το ODM μετά-μοντέλο.

Η KB είναι συμβατή με το JMI (Java Metadata Interface) 1.0 πρότυπο το οποίο είναι αποτέλεσμα της προσπάθειας της Java κοινότητας (Java Community Process, JCP) να δημιουργήσει ένα πρότυπο Java API για την πρόσβαση και την διαχείριση μετά-δεδομένων. Τα πλεονεκτήματα της συμβατότητας με το JMI 1.0 πρότυπο είναι τα ακόλουθα:

- Ο εφοδιασμός της Java 2 πλατφόρμας με ένα πρότυπο API διαχείρισης μετά-δεδομένων.
- Ο προσδιορισμός μίας τυπικής (formal) αντιστοίχισης (mapping) ενός οποιουδήποτε μετά-μοντέλου συμβατού με το OMG πρότυπο σε Java interfaces.
- Η υποστήριξη προχωρημένων (advanced) υπηρεσιών μετά-δεδομένων (όπως είναι ο δυναμικός προγραμματισμός).
- Η συνεργασία μεταξύ εργαλείων τα οποία βασίζονται σε MOF μετά-μοντέλα και εγκαθίστανται στο DBE περιβάλλον.

Έχει υλοποιηθεί ένα interface επικοινωνίας με την Knowledge Base, το Knowledge Base Interface (KBI) το οποίο χρησιμοποιήθηκε στις υλοποιήσεις των λειτουργιών της αποθήκευσης οντολογιών υλοποιημένων με το GRAMOFONE στην KB, της ανάκτησης οντολογιών από την KB στην επιφάνεια εργασίας του GRAMOFONE και της διαγραφής οντολογιών αποθηκευμένων στην KB.

**Ανακεφαλαίωση**

Στο κεφάλαιο περιγράφηκαν οι τεχνολογίες που μελετήθηκαν και χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής, δηλαδή η πλατφόρμα Eclipse, τα πλαίσια εργασίας Standard Widget Toolkit (SWT), JFace, Graphical Editing Framework (GEF), καθώς επίσης και η βάση γνώσης του DBE.

Κεφάλαιο 4

# Η ΑΡΧΙΤΕΚΤΟΝΙΚΗ MOF ΚΑΙ ΤΟ ΜΕΤΑ-ΜΟΝΤΕΛΟ ΟΡΙΣΜΟΥ ΟΝΤΟΛΟΓΙΩΝ (ODM)

## Εισαγωγή

Σε αυτό το κεφάλαιο θα παρουσιαστεί το πρότυπο MOF με το οποίο είναι συμβατό τόσο το μετά-μοντέλο ορισμού οντολογιών (ODM) όσο και το εργαλείο της παρούσας διπλωματικής διατριβής, και στη συνέχεια θα παρουσιαστεί η αρχιτεκτονική και η περιγραφή του μετά-μοντέλου ορισμού οντολογιών (ODM).

## Το MOF και η Αρχιτεκτονική Μετά-δεδομένων Τεσσάρων Επιπέδων

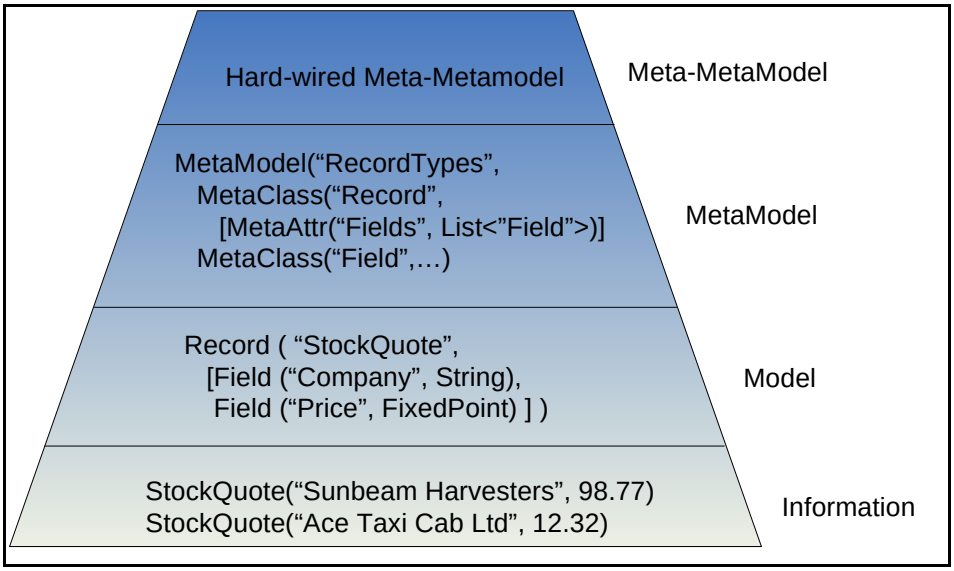
Σύμφωνα με τον διεθνή οργανισμό τυποποίησης OMG (Object Management Group), "το κεντρικό θέμα της τεχνολογίας του MOF είναι η διαχείριση μετά-δεδομένων με τρόπο που να εξασφαλίζει απεριόριστη επεκτασιμότητα. Σκοπός του είναι να παράσχει ένα πλαίσιο το οποίο θα υποστηρίζει κάθε είδους μετά-δεδομένα, και θα επιτρέπει την προσθήκη νέων ειδών όταν κρίνεται αναγκαίο. Για να επιτευχθεί αυτό, το MOF υιοθετεί μία πολύ-επίπεδη αρχιτεκτονική μετά-δεδομένων, η οποία βασίζεται στην κλασική αρχιτεκτονική μετά-δεδομένων τεσσάρων επιπέδων που είναι αρκετά δημοφιλής και χρησιμοποιείται σε πρότυπα όπως είναι το ISO και το CDIF". Η παραδοσιακή αρχιτεκτονική μετά-δεδομένων τεσσάρων επιπέδων εισάγει τα ακόλουθα επίπεδα:

1. Το επίπεδο πληροφορίας (Information layer). Αποτελείται από τα δεδομένα που επιθυμούμε να περιγράψουμε.
2. Το επίπεδο του Μοντέλου (Model layer). Περιέχει τα μετά-δεδομένα τα οποία περιγράφουν τα δεδομένα στο επίπεδο πληροφορίας. Τα μετά-δεδομένα συναθροίζονται ανεπίσημα σε μοντέλα.
3. Το επίπεδο του μετά-μοντέλου (Metamodel layer). Στο επίπεδο αυτό υπάρχουν οι περιγραφές οι οποίες προσδιορίζουν την δομή και την σημασιολογία των μετά-δεδομένων. Οι περιγραφές αυτές λέγονται μετά-μετά-δεδομένα (meta-metadata) και συναθροίζονται ανεπίσημα σε μετά-μοντέλα. Ένα μετά-μοντέλο είναι ουσιαστικά μία

"αφηρημένη γλώσσα" (η δομή μιας γλώσσας) περιγραφής διαφορετικών ειδών μετά-δεδομένων.

4. Το επίπεδο του μετά-μετά-μοντέλου (Meta-metamodel layer). Στο επίπεδο αυτό υπάρχει η περιγραφή (μία και μόνη) της δομής και της σημασιολογίας των μετά-μετά-δεδομένων (meta-metadata). Με άλλα λόγια είναι η "αφηρημένη γλώσσα" προσδιορισμού διαφορετικών ειδών μετά-μετά-δεδομένων (γλωσσών). [2]

Για την καλύτερη κατανόηση των εννοιών που περιγράφηκαν πιο πάνω, παρατίθεται το παράδειγμα της εικόνας 9 το οποίο παρέχεται από τον οργανισμό OMG και παρουσιάζει την κλασική αρχιτεκτονική τεσσάρων επιπέδων δείχνοντας την οργάνωση των μετά-δεδομένων για κάποιες απλές καταχωρήσεις μαζί με το "RecordTypes" μετά-μοντέλο το οποίο χρησιμοποιείται στον προσδιορισμό της δομής και της σημασιολογίας των μετά-δεδομένων.[2]




**Εικόνα 9: Η Αρχιτεκτονική Μετά-δεδομένων Τεσσάρων Επιπέδων**

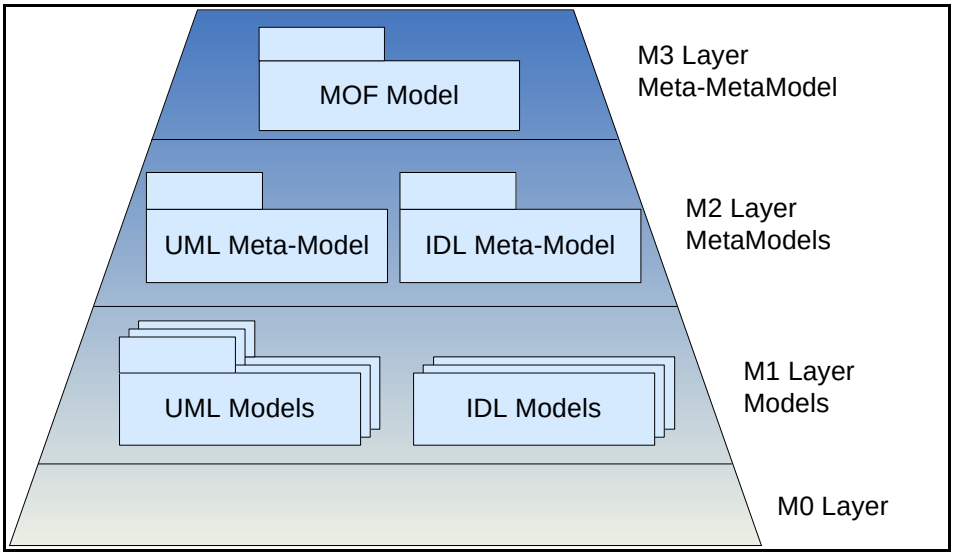
Ενώ το παράδειγμα της εικόνας 9 δείχνει μόνο ένα μοντέλο και ένα μετά-μοντέλο, ο βασικός σκοπός της ύπαρξης τεσσάρων (μετά-) επιπέδων είναι η υποστήριξη πολλαπλών μοντέλων και μετά-μοντέλων.

Σύμφωνα με τον οργανισμό OMG, "…Η κλασική αρχιτεκτονική μετά-δεδομένων τεσσάρων επιπέδων έχει αρκετά πλεονεκτήματα σε σχέση με τις απλές προσεγγίσεις μοντελοποίησης. Εάν το πλαίσιο μοντελοποίησης βασισμένο σε αυτή την αρχιτεκτονική σχεδιαστεί κατάλληλα, τότε:

- μπορεί να υποστηρίξει κάθε είδους μοντέλου και παραδείγματος μοντελοποίησης το οποίο είναι πρακτικά κατανοητό,
- μπορεί να επιτρέψει την συσχέτιση διαφορετικών ειδών μετά-δεδομένων,

- μπορεί να επιτρέψει την διαδοχική προσθήκη μετά-μοντέλων και νέων ειδών μετά-δεδομένων, και
- μπορεί να υποστηρίξει την ανταλλαγή αυθαίρετων μετά-δεδομένων (μοντέλα) και μετά-μετά-δεδομένων (μετά-μοντέλων) μεταξύ ομάδων οι οποίες χρησιμοποιούν το ίδιο μετά-μετά-μοντέλο."

Η MOF αρχιτεκτονική μετά-δεδομένων που φαίνεται στην εικόνα 10 βασίζεται στην αρχιτεκτονική που μόλις περιγράφηκε. Πρόκειται για ένα τυπικό παράδειγμα χρήσης της MOF αρχιτεκτονικής μετά-δεδομένων με μετά-μοντέλα για την αναπαράσταση των γλωσσών UML και IDL του οργανισμού OMG.




**Εικόνα 10: Η MOF Αρχιτεκτονική Μετά-δεδομένων**

Από την παραπάνω περιγραφή έχει γίνει σαφές ότι για να μπορούμε να διαχειριστούμε στο DBE οντολογίες (όπως έχουν οριστεί από την Semantic Web Community) χρειάζεται να προσδιοριστεί ένα μετά-μοντέλο περιγραφής οντολογιών (ODM) χρησιμοποιώντας το MOF πρότυπο. Πρέπει να τονιστεί ότι ο κεντρικός μηχανισμός μοντελοποίησης στην αρχιτεκτονική που παρουσιάστηκε πιο πάνω είναι το MOF (Meta-Object Facility) το οποίο παρέχει τις βασικές έννοιες (primitives) για την δημιουργία μετά-μοντέλων.

Το MOF ορίζει τέσσερις βασικές έννοιες (primitives) μοντελοποίησης οι οποίες θα παρουσιαστούν παρακάτω:

- **Κλάσεις (Classes),** οι οποίες χρησιμοποιούνται για την μοντελοποίηση των MOF μετά-αντικειμένων. Οι κλάσεις μπορούν να έχουν τριών ειδών ιδιότητες:
  - Χαρακτηριστικά (Attributes).
  - Λειτουργίες (Operations).
  - Αναφορές (References) μεταξύ κλάσεων.

Οι κλάσεις μπορούν επιπλέον να περιέχουν Εξαιρέσεις (Exceptions) που μπορούν να προκληθούν από Λειτουργίες (Operations), Σταθερές (Constants), Τύπους Δεδομένων (DataTypes) όπως για παράδειγμα Boolean, Integer, String, Enumerations κλπ., Περιορισμούς (Constraints) όπως για παράδειγμα τον περιορισμό πως ένα Attribute "x:integer" είναι μονός αριθμός , και άλλα στοιχεία. Επιπλέον επιτρέπεται η κληρονομικότητα μεταξύ των κλάσεων, ενώ μία κλάση μπορεί να οριστεί σαν «αφηρημένη» (abstract). Τέλος μία κλάση μπορεί να οριστεί είτε σαν «φύλλο» (leaf), οπότε δεν μπορεί να έχει υπό-κλάσεις, είτε σαν «ρίζα» (root), οπότε δεν μπορεί να έχει υπέρ-κλάσεις.

- **Συσχετίσεις (Associations)**, οι οποίες χρησιμοποιούνται για την μοντελοποίηση των σχέσεων μεταξύ δύο μετά-αντικειμένων (Κλάσεων).
- **Τύποι Δεδομένων (DataTypes)**, οι οποίοι χρησιμοποιούνται για την μοντελοποίηση άλλων δεδομένων (π.χ. primitive τύποι, εξωτερικοί τύποι κτλ.).
- **Πακέτα (Packages)**, τα οποία χρησιμοποιούνται για την ομαδοποίηση των μοντέλων.

Δεν θα προχωρήσουμε σε περαιτέρω ανάλυση των εννοιών που ορίζονται από το MOF καθώς η βαθύτερη ανάλυσή τους δεν περιέχεται στο πλαίσιο της παρούσας διπλωματικής διατριβής. Η πλήρης ανάλυση του MOF (Meta Object Facility) δίνεται στο «Meta Object Facility (MOF) Specification», το οποίο είναι διαθέσιμο από την επίσημη σελίδα στο Διαδίκτυο του οργανισμού Object Management Group (OMG). [24]

**Μετά-μοντέλο Ορισμού Οντολογιών - Ontology Definition Metamodel (ODM)**

Το Ontology Definition Metamodel (ODM) είναι ένα μετά-μοντέλο υλοποιημένο στο πλαίσιο του προγράμματος DBE με στόχο τον προσδιορισμό business και service οντολογιών οι οποίες θα διευκολύνουν την πραγματοποίηση του διαμοιρασμού της γνώσης μεταξύ επιχειρήσεων (SME: Small and Medium sized Enterprise) και θα κάνουν δυνατή την ενσωμάτωση των υπηρεσιών των επιχειρήσεων αυτών από τεχνικής άποψης. Το ODM θα πρέπει να είναι σε θέση να αναπαριστά υπάρχουσες οντολογίες περιγεγραμμένες με κάποια ήδη αποδεκτή γλώσσα περιγραφής οντολογιών για λόγους χρηστικότητας και συμβατότητας. Η πλειοψηφούσα γνώμη μέσα στην κοινότητα που ασχολείται με τις οντολογίες είναι πως η Ontology Web Language (OWL) είναι η επικρατέστερη προσπάθεια ως προς την τυποποίηση των γλωσσών περιγραφής οντολογιών και αναμένεται ότι πολλές οντολογίες θα περιγραφούν στον μέλλον με την OWL και πολλοί άνθρωποι θα γνωρίζουν την OWL ώστε να είναι σε θέση να καταλαβαίνουν εύκολα οντολογίες προσδιορισμένες με την OWL αλλά και να είναι σε θέση να γράψουν οντολογίες σε αυτή την γλώσσα. Το ODM

είναι ο πυρήνας του Μοντέλου Αναπαράστασης Γνώσης (Knowledge Representation Model) του DBE καθώς μπορεί να θεωρηθεί σαν ο γενικός μηχανισμός αναπαράστασης της γνώσης.[2]

Συγκεκριμένα οι στόχοι του ODM είναι οι εξής:

- Η δημιουργία ενός μετά-μοντέλου περιγραφής οντολογιών συμβατό με το πρότυπο MOF 1.4.
- Η δυνατότητα αναπαράστασης οντολογιών που υπάρχουν ήδη και είναι περιγεγραμμένες με την OWL DL.
- Η υλοποίηση μιας γλώσσας η οποία να αντιστοιχίζει όρους του ODM με όρους της OWL DL.
- Ο προσδιορισμός ενός UML 1.4 προφίλ για την επαναχρησιμοποίηση UML όρων στην διαδικασία μοντελοποίησης με το ODM (η παρούσα έκδοση δεν καλύπτει τον στόχο αυτό).

Ο οργανισμός Object Management Group (OMG) έθεσε μία πρόσκληση για προτάσεις (RFP: Request For Proposals) για τον καθορισμό ενός μετά-μοντέλου ορισμού οντολογιών (ονομαζόμενο ODM) ο οποίος να βασίζεται στο MOF πρότυπο. Ο οργανισμός OMG πιστεύει ότι η εξοικείωση των χρηστών με την UML, η διαθεσιμότητα UML εργαλείων, η ύπαρξη πολλών μοντέλων σε UML, και η ομοιότητα αυτών των μοντέλων με τις οντολογίες, υποδηλώνει ότι η UML μπορεί να είναι ένα μέσο προς την ταχύτερη ανάπτυξη των οντολογιών. Υποδηλώνει λοιπόν ότι πρέπει να δημιουργηθεί ένα UML προφίλ το οποίο θα δίνει την δυνατότητα στους σχεδιαστές να χρησιμοποιούν το ODM με παρόμοιο τρόπο. Επιπλέον, υποδηλώνει ότι πρέπει να υπάρξει και μία αντιστοίχιση μεταξύ του ODM και της OWL. Το υλοποιημένο ODM μετά-μοντέλο έχει αναπτυχθεί έτσι ώστε να ικανοποιεί όλες τις ανάγκες που θέτει ο οργανισμός OMG. Ωστόσο, δεν είναι εξασφαλισμένο ότι το υλοποιημένο ODM θα είναι συμβατό με το πρότυπο το οποίο θα βγάλει τελικά η OMG. Επειδή όμως το ODM καλύπτει τους στόχους που θέτονται από την OMG, πιστεύεται πως τα δύο μετά-μοντέλα θα είναι σε πολύ μεγάλο βαθμό συμβατά.

Στο DBE οραματίζονται την ευρεία χρήση των οντολογιών σαν τον κύριο μηχανισμό αναπαράστασης, τυποποίησης και διαμοιρασμού της γνώσης. Συγκεκριμένα, σε κάθε επιχειρηματικό τομέα, οντολογίες εξειδικευμένες για τον τομέα αυτό θα πρέπει να παρέχονται από ειδικούς του τομέα και να χρησιμοποιούνται σαν σημείο αναφοράς από τις επιχειρήσεις που ανήκουν στον τομέα αυτό. Επιπλέον, κάθε επιχείρηση θα πρέπει να έχει την δυνατότητα να επεκτείνει αυτές τις οντολογίες ή να δημιουργεί τις δικές της με σκοπό να περιγράψει επαγγελματικούς όρους οι οποίοι δεν καλύπτονται από τις γενικές οντολογίες του

τομέα. Και στις δύο περιπτώσεις, (είτε ο δημιουργός είναι ειδικός ή είναι μία επιχείρηση) η προσδιορισμός των οντολογιών μπορεί να γίνει μέσω του ODM.

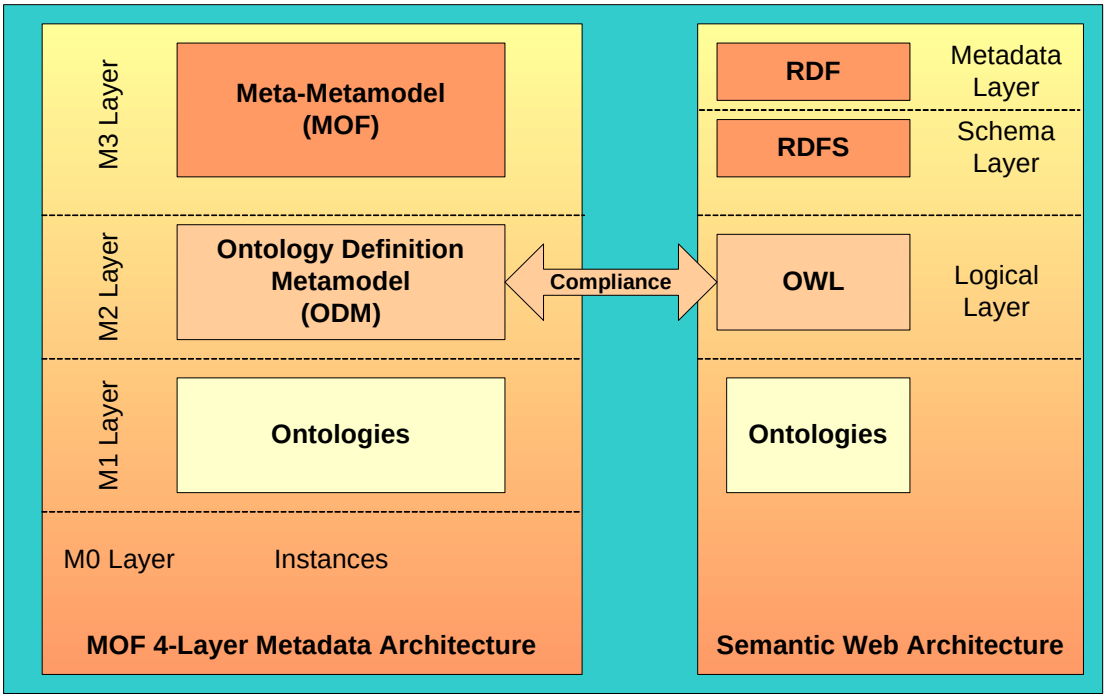
Η MOF αρχιτεκτονική χρησιμοποιεί μία αμετάβλητη αρχιτεκτονική μετά-μοντελοποίησης τεσσάρων επιπέδων για την διαχείριση των μετά-δεδομένων. Με τον όρο «αμετάβλητη» εννοούμε ότι υπάρχουν μόνο 2 επίπεδα κλάσεων και η μετάβαση από το ένα επίπεδο στο άλλο, έχει ξεκάθαρη σημασιολογία (instantiation). Από την άλλη πλευρά, η αρχιτεκτονική του Semantic Web είναι μεταβλητή, και προσδιορίζει τα δικά της επίπεδα με διαφορετική σημασιολογία από αυτή της αρχιτεκτονικής του MOF. Με τον όρο «μεταβλητή» εννοούμε ότι υπάρχει η δυνατότητα απεριορίστων επιπέδων κλάσεων καθώς η rdfs:class είναι ένα στιγμιότυπο του εαυτού της.

Μια άλλη διαφορά μεταξύ των δύο αρχιτεκτονικών είναι αυτή των στιγμιότυπων (instances). Στην αρχιτεκτονική του semantic web μια οντολογία (π.χ. μια OWL οντολογία) εμπεριέχει τόσο τις έννοιες (κλάσεις) που περιγράφουν τη γνώση ενός πεδίου, όσο και τα στιγμιότυπα (individuals) εκείνα τα οποία θεωρούνται μέρος της κοινά αποδεκτής γνώσης που αναπαριστά η οντολογία. Στην αρχιτεκτονική του MOF από την άλλη μεριά, τα στιγμιότυπα μιας κλάσης που ορίζει ο χρήστης σε ένα επίπεδο (π.χ. M1), υπάρχουν στο επόμενο επίπεδο (π.χ. M0). Προκειμένου λοιπόν το ODM να είναι πλήρως συμβατό με την OWL έπρεπε να δίνει τη δυνατότητα ορισμού κλάσεων και στιγμιότυπων στο ίδιο επίπεδο.

Το ODM είναι ένα μοντέλο επιπέδου M2 (δηλαδή ένα μετά-μοντέλο) στην αρχιτεκτονική του MOF. Τα στιγμιότυπα (instances) αυτού του μετά-μοντέλου (μοντέλα επιπέδου M1) είναι οι οντολογίες που δημιουργούνται από το χρήστη και μπορούν να περιλαμβάνουν τόσο κλάσεις όσο και στιγμιότυπα των κλάσεων στο ίδιο επίπεδο.

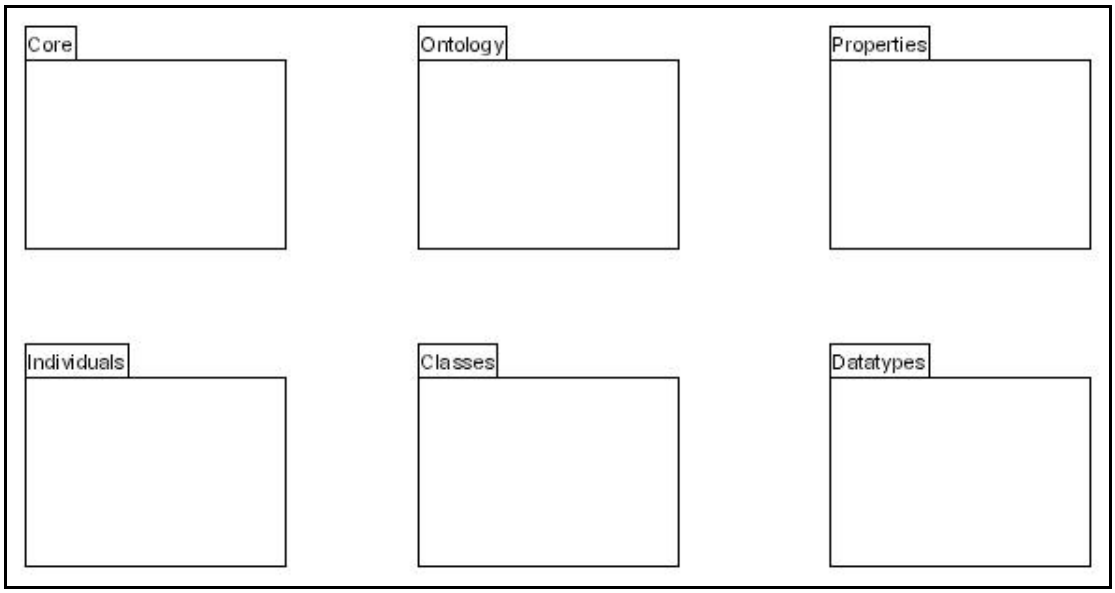
Επιπλέον σε επίπεδο στιγμιότυπων, μία rdfs:class προσδιορίζεται από τα στιγμιότυπά της, ενώ μία MOF Class προσδιορίζει τα στιγμιότυπά της. Επίσης στην RDFS ένα στιγμιότυπο μπορεί να είναι μέλος (instance) μίας ή περισσοτέρων κλάσεων, ενώ στην αρχιτεκτονική του MOF κάθε στιγμιότυπο πρέπει να έχει ακριβώς μία κλάση.

Η εικόνα 11 δείχνει το ODM στο πλαίσιο της αρχιτεκτονικής του MOF και την αντιστοίχισή του με την αρχιτεκτονική του Semantic Web. [2]




**Εικόνα 11: Αρχιτεκτονική Προσέγγιση του ODM**

Το ODM αποτελείται από πακέτα (packages) καθένα από τα οποία προσδιορίζει μία συγκεκριμένη πτυχή του μετά-μοντέλου. Η εικόνα 12 δείχνει τα πακέτα από τα οποία αποτελείται το μετά-μοντέλο.




**Εικόνα 12: Τα Πακέτα του ODM**

- Το πακέτο "Core" είναι ένα από τα πιο σημαντικά πακέτα του μετά-μοντέλου και προσδιορίζει τις αφηρημένες έννοιες που χρησιμοποιούνται σαν υπέρ-κλάσεις (super-classes) όλων των υπολοίπων όρων στο μετά-μοντέλο.

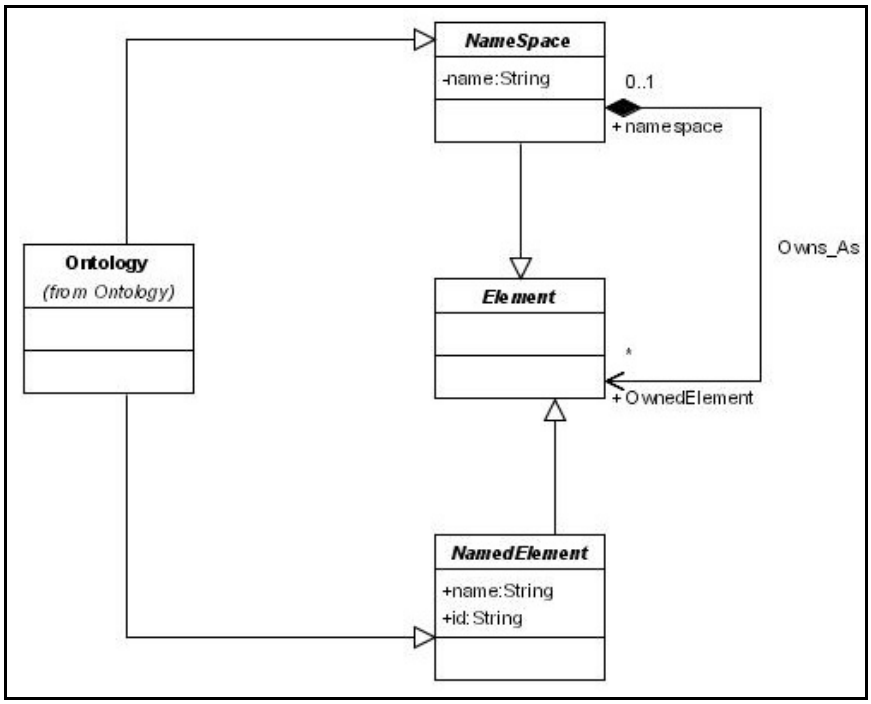
- Το πακέτο "DataTypes" ορίζει επίσης ένα θεμελιώδες τμήμα του μετά-μοντέλου καθώς εισάγει τον όρο του τύπου δεδομένων (data type) και τους σχετικούς με αυτόν όρους.
- Το πακέτο "Ontology" ορίζει τι είναι μία οντολογία και πως συσχετίζεται με άλλους όρους του μετά-μοντέλου.
- Η έννοια της κλάσης (class) και των σχετικών με αυτήν όρων (π.χ. αξιώματα κλάσεων, τρόποι ορισμού μιας κλάσης, σχόλια κτλ.) ορίζονται στο πακέτο "Classes".
- Το πακέτο "Properties" ορίζει διάφορους όρους σχετικούς με τις ιδιότητες των κλάσεων.
- Το πακέτο "Individuals" ορίζει πως τα στιγμιότυπα των όρων μίας οντολογίας περιγράφονται και συσχετίζονται μεταξύ τους.

Στη συνέχεια θα προχωρήσουμε στην περιγραφή του ίδιου του μετά-μοντέλου και των εννοιών που ορίζονται μέσα σε αυτό.

**Αφηρημένες Έννοιες (Abstract Concepts) Στο ODM**

Όπως αναφέρθηκε προηγουμένως το Core πακέτο ορίζει της αφηρημένες μετά-κλάσεις (από εδώ και στο εξής κλάσεις) οι οποίες χρησιμοποιούνται σαν υπέρ-κλάσεις όλων των άλλων κλάσεων του ODM.

Η αρχική (root) κλάση στο ODM είναι η κλάση Element. Η Element είναι η αφηρημένη κλάση (abstract class) η οποία είναι κοινή υπέρ-κλάση όλων των άλλων κλάσεων στο ODM. Σε μια οντολογία υπάρχουν στοιχεία που δεν έχουν όνομα (π.χ. τιμές) και κάποια τα οποία προσδιορίζονται από κάποιο όνομα. Κάθε στοιχείο το οποίο μπορεί να οριστεί σε μία οντολογία και θα πρέπει να περιγράφεται με ένα όνομα, μοντελοποιείται μέσω της αφηρημένης κλάσης NamedElement η οποία είναι υπό-κλάση της κλάσης Element. Τα NamedElements αναγνωρίζονται επίσης από ένα id το οποίο είναι μοναδικό σε ένα καθορισμένο namespace. Η αφηρημένη κλάση NameSpace χρησιμοποιείται για την ομαδοποίηση (scoping) των στοιχείων μοντελοποίησης. Το namespace είναι ένα ειδικό είδος στοιχείου (ένα container) το οποίο περιέχει άλλα στοιχεία. Για τον λόγο αυτό, κάθε στοιχείο σε μία οντολογία ανήκει σε ένα (το πολύ) namespace. Στην παρούσα έκδοση του μετά-μοντέλου μόνο μία οντολογία μπορεί να χρησιμοποιηθεί σαν namespace για άλλα στοιχεία (τους όρους της που εσωκλείει). Στην εικόνα 13 παρουσιάζονται οι έννοιες που περιγράφηκαν πιο πάνω.




**Εικόνα 13: Αφηρημένες Έννοιες (Abstract Concepts) Στο ODM**

Από το διάγραμμα της εικόνας 13 μπορούμε να συμπεράνουμε πως:

a) μία οντολογία μπορεί να περιέχει άλλες οντολογίες, και
b) ένα στοιχείο μπορεί να έχει ή μπορεί να μην έχει namespace.

Κανένα από τα παραπάνω συμπεράσματα δεν πρέπει να ισχύει, και το εργαλείο που θα χρησιμοποιείται για τη δημιουργία οντολογιών με το ODM θα πρέπει να εξασφαλίζει σωστή μοντελοποίηση αποτρέποντας την εμφάνιση τέτοιων καταστάσεων.

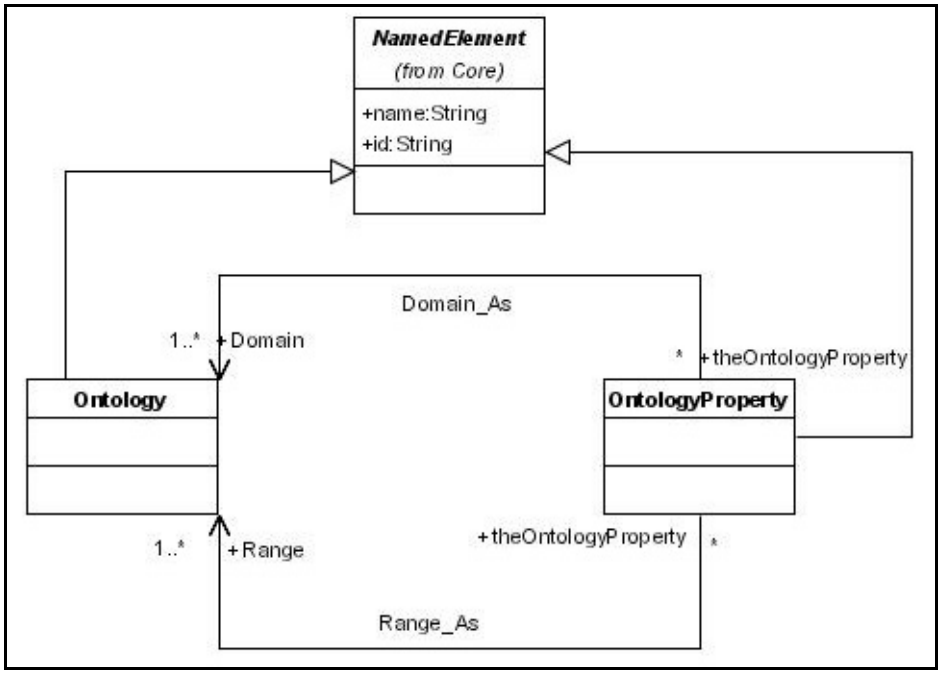
Αν και μία οντολογία δεν μπορεί να περιέχει άλλες οντολογίες, μπορεί να εισάγει (import) άλλες οντολογίες κάνοντας χρήση της "imports" σχέσης που περιγράφεται παρακάτω. Το ODM δεν επιτρέπει στις έννοιες μίας οντολογίας (κλάσεις (classes), ιδιότητες (properties), στιγμιότυπα (things) κτλ.) να ορίζονται χωρίς κάποιο καθορισμένο namespace. Μόνο μία οντολογία μπορεί να οριστεί χωρίς να έχει συσχετιστεί με κάποιο namespace (μιας και η ίδια αποτελεί namespace για τα στοιχεία που περικλείει).

### Οντολογίες (Ontologies)

Μία οντολογία προσδιορίζει τα διάφορα στοιχεία τα οποία μπορούν να χρησιμοποιηθούν στην περιγραφή και την αναπαράσταση ενός πεδίου γνώσης. Περιλαμβάνει προσδιορισμούς βασικών εννοιών ενός πεδίου γνώσης και των μεταξύ τους σχέσεων, χρησιμοποιώντας στοιχεία όπως είναι οι κλάσεις (classes), τα στιγμιότυπα (individuals), οι ιδιότητες (properties) κτλ. όπως αυτά ορίζονται στο ODM. Όπως φαίνεται στην εικόνα 13, μία οντολογία αποτελεί ένα namespace για τα στοιχεία που εσωκλείει. Γενικεύοντας, μία

οντολογία περιέχει ορισμούς και ιεραρχίες εννοιών (π.χ. κλάσεις). Παραδείγματα τέτοιων ορισμών είναι ο ορισμός της έννοιας "άτομο", ο ορισμός ιδιοτήτων (properties) εννοιών (π.χ. το όνομα ή η ηλικία του ατόμου), ο ορισμός σχέσεων μεταξύ εννοιών (π.χ. ένα άτομο *"έχει_φίλο"* κάποιο άλλο άτομο), και ο ορισμός στιγμιότυπων (things ή individuals) εννοιών (π.χ. ο *"Γιάννης"* ο οποίος έχει_φίλο τον *"Νίκο"*). Αυτές είναι οι βασικές έννοιες της περιγραφής μίας οντολογίας. Βασισμένες σε αυτές τις έννοιες μοντελοποίησης μπορούν να οριστούν άλλες δομές όπως είναι οι περιορισμοί (restrictions), οι λίστες (lists) κτλ.

Στην Semantic Web κοινότητα, οι οντολογίες έχουν συγκεκριμένες ιδιότητες οι οποίες χρησιμοποιούνται για σημασιολογικούς σκοπούς. Τέτοιες ιδιότητες μοντελοποιούνται στο ODM μέσω της έννοιας OntologyProperty. Πιο συγκεκριμένα, μία OntologyProperty συσχετίζει μία οντολογία με άλλες οντολογίες. Ένα πλήθος δομών (π.χ. η δομή προσδιορισμού της έκδοσης (version) μίας οντολογίας), μπορούν να οριστούν σαν στιγμιότυπα της έννοιας OntologyProperty. Τα στιγμιότυπα αυτά πρέπει να συσχετιστούν με συγκεκριμένες οντολογίες (στιγμιότυπα της έννοιας Οντολογία (Ontology)). Όπως φαίνεται στην εικόνα 14 ένα OntologyProperty στιγμιότυπο πρέπει να έχει τουλάχιστον μία οντολογία ορισμένη σαν domain (συνήθως η οντολογία που επεξεργαζόμαστε) και τουλάχιστον μία οντολογία ορισμένη σαν range. Παραδείγματα τέτοιων στιγμιότυπων (της OntologyProperty) είναι οι ιδιότητες "imports", "priorVersion", "backwardCompatibleWith" και "incompatibleWith" οι οποίες ορίζουν συγκεκριμένες σχέσεις μεταξύ οντολογιών.




**Εικόνα 14: Οντολογία (Ontology) και Ιδιότητες Οντολογιών (Ontology Properties)**

**Κλάσεις (Classes)**

Μία θεμελιώδης έννοια στο ODM είναι η κλάση (class). Η κλάση αναπαριστά την ομαδοποίηση παρόμοιων στοιχείων (που ονομάζονται στιγμιότυπα (things ή individuals)) μέσα σε μία οντολογία. Για τον λόγο αυτό κάθε κλάση συνδέεται με ένα σύνολο από στιγμιότυπα, που ονομάζεται επέκταση της κλάσης (class extension). Για παράδειγμα σε μία οντολογία μπορεί να οριστεί η κλάση "Χρώμα_Κρασιού", η οποία να αναπαριστά την ομαδοποίηση των στιγμιότυπων "Κόκκινο", "Άσπρο" και "Ροζέ". Είναι εύκολο να καταλάβει κανείς πως στο προηγούμενο παράδειγμα πρώτα δημιουργείται η κλάση και στην συνέχεια τα στιγμιότυπά της. Η σημασιολογία μίας κλάσης έχει σχέση αλλά δεν ταυτίζεται με την επέκτασή της (class extension). Αυτό σημαίνει πως δύο κλάσεις μπορεί ενώ έχουν την ίδια επέκταση (class extension) να είναι διαφορετικές η μία από την άλλη σημασιολογικά.

**Προσδιορισμοί Κλάσεων (Class Definitions) στο ODM**

Στο ODM μία κλάση μπορεί να οριστεί είτε μέσω ενός ονόματος (class name) είτε προσδιορίζοντας την επέκταση (class extension) μίας ανώνυμης κλάσης. Όπως ορίζεται από την OWL, το ODM διακρίνει έξι τύπους προσδιορισμού μίας κλάσης (class definitions):

i. Σαν μία καινούργια κλάση, ορίζοντας ένα όνομα (class name) και ένα αναγνωριστικό (id).

ii. Σαν μία απαρίθμηση (enumeration) από στιγμιότυπα (individuals) τα οποία αποτελούν τα στιγμιότυπα της κλάσης. Όπως σημειώθηκε πιο πάνω, μία κλάση μπορεί να οριστεί σαν ένα σύνολο από στιγμιότυπα (individuals). Αυτός ο ορισμός πραγματοποιείται μέσω της σχέσης «oneOf» της εικόνας 15. Η επέκταση της κλάσης (class extension) σε αυτή την περίπτωση είναι όλα τα στιγμιότυπα τα οποία συνδέονται με την κλάση μέσω της σχέσης «oneOf». Για παράδειγμα η κλάση "Χρώμα_Κρασιού" σε μία οντολογία οινολογίας μπορεί να οριστεί σαν ένα από τα "Κόκκινο", "Άσπρο" ή "Ροζέ".

iii. Σαν έναν περιορισμό σε μία ιδιότητα (property restriction). Μία λεπτομερής περιγραφή αυτού του ορισμού θα δοθεί παρακάτω.

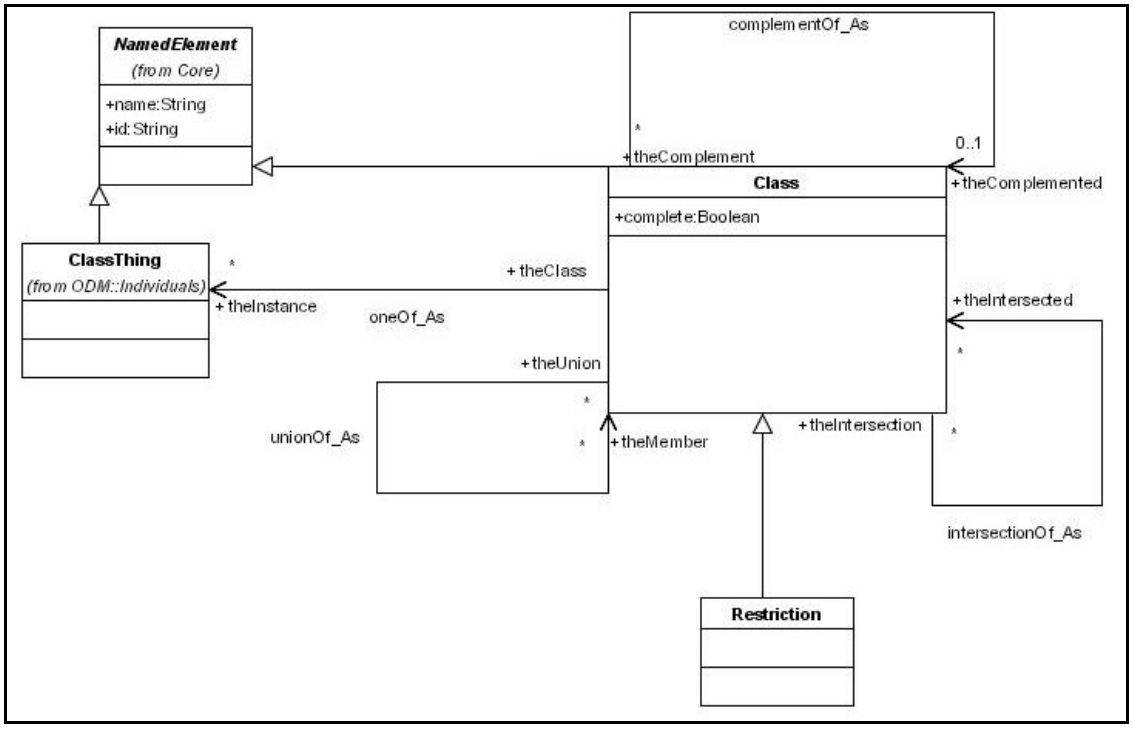
iv. Σαν η τομή (intersection) δύο ή περισσοτέρων ορισμών κλάσεων (class definitions). Ουσιαστικά ο ορισμός αυτός αντιστοιχεί στον τελεστή «AND» ο οποίος εφαρμόζεται στους σχετικούς ορισμούς των κλάσεων (class definitions). Ο ορισμός αυτός πραγματοποιείται μέσω της σχέσης «intersectionOf» της εικόνας 15. Για παράδειγμα, η κλάση "Λευκό_Επιτραπέζιο_Κρασί" σε μία οντολογία οινολογίας

μπορεί να οριστεί σαν η τομή των κλάσεων "Λευκό_Κρασί" και "Επιτραπέζιο_Κρασί".

v. Σαν η ένωση (union) δύο ή περισσοτέρων ορισμών κλάσεων (class definitions). Ο ορισμός αυτός αντιστοιχεί στον τελεστή «OR» ο οποίος εφαρμόζεται στους σχετικούς ορισμούς των κλάσεων (class definitions). Ο ορισμός αυτός πραγματοποιείται μέσω της σχέσης «unionOf» της εικόνας 15. Για παράδειγμα, η κλάση "Περιγραφή_Κρασιού" σε μία οντολογία οινολογίας μπορεί να οριστεί σαν την ένωση των κλάσεων "Χρώμα_Κρασιού" και "Γεύση_Κρασιού".

vi. Σαν το συμπλήρωμα (complement) ενός ορισμού μίας κλάσης (class definition). Ο ορισμός αυτός αντιστοιχεί στον τελεστή «NOT» ο οποίος εφαρμόζεται στον ορισμό της κλάσης (class definition). Ο ορισμός αυτός πραγματοποιείται μέσω της σχέσης «complementOf» της εικόνας 15. Για παράδειγμα, η κλάση "Εργαζόμενος_Πλήρους_Απασχόλησης" μπορεί να οριστεί σαν το συμπλήρωμα της κλάσης "Εργαζόμενος_Μερικής_Απασχόλησης" σε μία οντολογία περιγραφής μίας εταιρίας και των εργαζομένων σε αυτήν.

Ο πρώτος τύπος προσδιορισμού μίας κλάσης είναι ειδικός από την άποψη πως ορίζει μία αυτούσια (standalone) έννοια, μία έννοια δηλαδή που μπορεί να σταθεί από μόνη της, έχει δική της υπόσταση. Οι υπόλοιποι πέντε τύποι προσδιορισμού κλάσεων περιγράφουν μία ανώνυμη κλάση θέτοντας περιορισμούς στην επέκταση (class extension) άλλων κλάσεων.

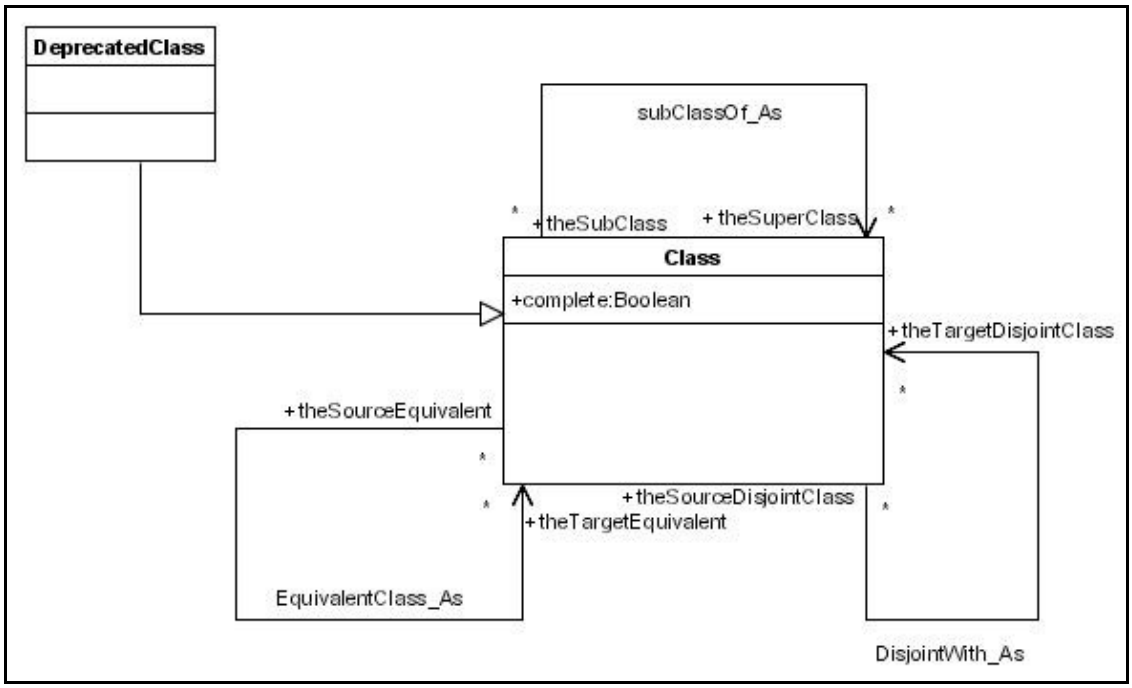



**Εικόνα 15: Προσδιορισμοί Κλάσεων (Class Definitions) στο ODM**

**Αξιώματα Κλάσεων (Class Axioms) στο ODM**

Στην διάρκεια υλοποίησης μίας οντολογίας, ο προσδιορισμός των κλάσεων μπορεί να είναι ξεχωριστός ή συσχετιζόμενος με άλλες κλάσεις. Για παράδειγμα η κλάση "Λευκό_Κρασί" μίας οντολογίας οινολογίας θα μπορούσε να οριστεί ξεχωριστά από τις άλλες κλάσεις της οντολογίας. Αν και είναι συντακτικά σωστό σύμφωνα με το ODM, δεν δίνει πολλές πληροφορίες σχετικά με την έννοια "Λευκό_Κρασί". Εάν για παράδειγμα γνωρίζαμε πως η κλάση "Λευκό_Κρασί" είναι ένας ειδικός τύπος της κλάσης "Κρασί", αυτό θα ήταν πιο χρήσιμο και ακριβές για το συγκεκριμένο πεδίο γνώσης. Για την αναπαράσταση τέτοιων δεδομένων, χρειάζεται να οριστούν ειδικά αξιώματα (axioms), δηλαδή δομές οι οποίες να συσχετίζουν τις κλάσεις μεταξύ τους. Όπως ορίζεται από την OWL, το ODM ορίζει συγκεκριμένες δομές με σκοπό την παροχή ενός μηχανισμού για την συσχέτιση κλάσεων που ορίζονται στις οντολογίες. Πιο συγκεκριμένα, ορίζει τρεις γλωσσικές δομές:

i. Η σχέση «subClassOf»: η σχέση επιτρέπει τον προσδιορισμό της επέκτασης (extension) του ορισμού μίας κλάσης σαν υποσύνολο της επέκτασης του ορισμού μίας άλλης κλάσης. Αυτός ο ορισμός υπονοεί ότι μία κλάση είναι υπό-κλάση του εαυτού της. Ωστόσο, καθώς μία τέτοια λειτουργία δεν έχει να προσφέρει τίποτα από σημασιολογικής απόψεως, το ODM δεν επιτρέπει στους ορισμούς των κλάσεων την δήλωση της κλάσης σαν υπό-κλάση του εαυτού της.
ii. Η σχέση «equivalentClass»: η σχέση επιτρέπει στον χρήστη να ορίσει πως ο ορισμός μίας κλάσης έχει ακριβώς την ίδια επέκταση (class extension) με τον ορισμό μίας άλλης κλάσης.
iii. Η σχέση «disjointWith»: η σχέση επιτρέπει στον χρήστη να ορίσει πως η επέκταση (class extension) του ορισμού μίας κλάσης δεν έχει κοινά μέλη με την επέκταση (class extension) του ορισμού μίας άλλης κλάσης.



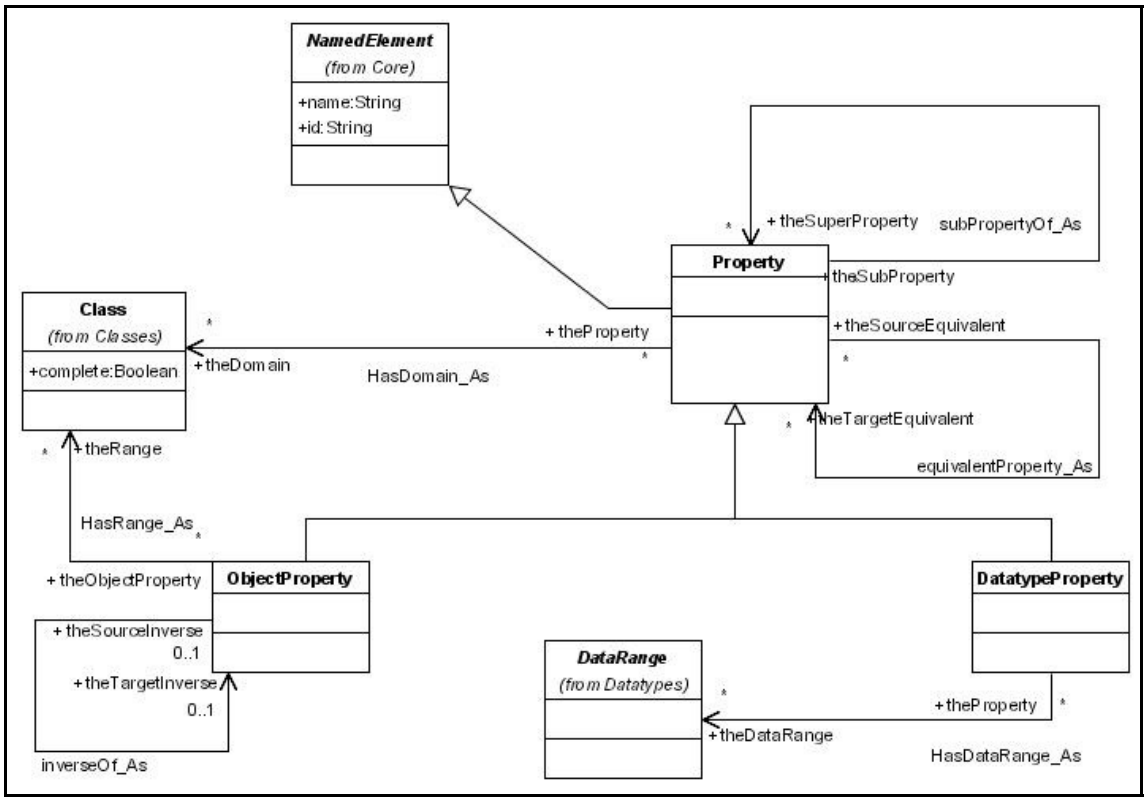

**Εικόνα 16: Αξιώματα Κλάσεων (Class Axioms) στο ODM**

### Ιδιότητες Κλάσεων (Class Properties) στο ODM

Μία κλάση χρησιμοποιείται στην περιγραφή εννοιών όπως η έννοια "άτομο", η έννοια "αυτοκίνητο" κτλ. για ένα συγκεκριμένο πεδίο γνώσης. Στον πραγματικό κόσμο, οι έννοιες έχουν συγκεκριμένα χαρακτηριστικά και μπορούν να συσχετίζονται με άλλες έννοιες. Για την αντιμετώπιση αυτής της ανάγκης, το RDFS χρησιμοποιεί την έννοια «property» (ιδιότητα). Μία ιδιότητα μπορεί επίσης να θεωρηθεί σαν ένας μηχανισμός για την συσχέτιση συγκεκριμένων στιγμιότυπων με άλλα στιγμιότυπα ή με συγκεκριμένες τιμές δεδομένων. Για παράδειγμα, το στιγμιότυπο "ChateauMargaux" (στιγμιότυπο της κλάσης "Margaux", η οποία είναι ποικιλία κρασιού) "παρασκευάζεται_από" (η ιδιότητα) το στιγμιότυπο "ChateauMargauxWinery" (στιγμιότυπο της κλάσης "Winery" (οινοποιείο)) στο πλαίσιο μίας οντολογίας οινολογίας. Σε αυτό το παράδειγμα, η δήλωση "παρασκευάζεται_από" έχει οριστεί σαν ιδιότητα της κλάσης "Margaux" η οποία συνδέει ένα στιγμιότυπο της κλάσης "Margaux" με ένα στιγμιότυπο της κλάσης "Winery". Ας θεωρήσουμε τώρα ότι επιθυμούμε να περιγράψουμε ότι το στιγμιότυπο "Έτος_1998" (στιγμιότυπο της κλάσης "Έτος_Σοδειάς") "έχει_χρονιά" (η ιδιότητα) 1998 στο πλαίσιο της ίδιας οντολογίας οινολογίας. Σε αυτό το παράδειγμα, η δήλωση "έχει_χρονιά" έχει οριστεί σαν ιδιότητα της κλάσης "Έτος_Σοδειάς" η οποία συνδέει ένα στιγμιότυπο της κλάσης "Έτος_Σοδειάς" με ένα θετικό ακέραιο αριθμό.

Είναι σαφές ότι μία κλάση μπορεί να έχει δύο ειδών ιδιότητες (properties): ιδιότητες οι οποίες συνδέουν στιγμιότυπα μίας κλάσης με τα στιγμιότυπα μίας άλλης κλάσης, και

ιδιότητες που συνδέουν τα στιγμιότυπα μίας κλάσης με συγκεκριμένες τιμές δεδομένων (literals). Εάν θεωρήσουμε τις ιδιότητες σαν δηλώσεις οι οποίες συσχετίζουν υποκείμενα με αντικείμενα, τότε αυτό που διαφέρει σε αυτά τα δύο είδη ιδιοτήτων είναι το αντικείμενο. Ωστόσο, το υποκείμενο (domain) και των δύο ιδιοτήτων είναι το ίδιο (κλάση).




**Εικόνα 17: Ιδιότητες Κλάσεων (Class Properties) στο ODM**

Όσον αφορά το αντικείμενο (range) μίας ιδιότητας, το ODM ορίζει δύο είδη ιδιοτήτων όπως φαίνεται στην εικόνα 17:

i. Ιδιότητες μεταξύ κλάσεων (Object Property): Πρόκειται για την ιδιότητα η οποία έχει σαν αντικείμενο (range) μία άλλη κλάση. Αυτό το είδος ιδιοτήτων συσχετίζει στιγμιότυπα μίας κλάσης με τα στιγμιότυπα μιας άλλης κλάσης. Είναι δυνατό να οριστεί πως μία ιδιότητα αυτού του είδους (object property) είναι αντίστροφη κάποιας άλλης ιδιότητας του ίδιου είδους (object property). Ο ορισμός αυτός πραγματοποιείται μέσω της σχέσης «inverseOf_As» της εικόνας 17. Για παράδειγμα, η ιδιότητα "παρασκευάζεται_από_σταφύλι" της κλάσης "Κρασί" σε μία οντολογία οινολογίας, είναι αντίστροφη της ιδιότητας "μετατρέπεται_σε_κρασί" της κλάσης "Σταφύλι". Η σχέση «inverseOf» μεταξύ των ιδιοτήτων είναι συμμετρική. Αυτό σημαίνει πως αν η ιδιότητα A είναι αντίστροφη της ιδιότητας B, τότε η ιδιότητα B είναι αντίστροφη της ιδιότητας A.

ii. Ιδιότητες μεταξύ κλάσεων και συγκεκριμένων τιμών (Datatype Property): Πρόκειται για την ιδιότητα η οποία παίρνει συγκεκριμένες τιμές σαν αντικείμενο (range). Για τον σκοπό αυτό ορίζεται η έννοια του Εύρους Τιμών (DataRange). Όπως θα περιγραφεί παρακάτω, το Εύρους Τιμών είναι μία αφηρημένη δομή προσδιορισμού του γεγονότος ότι το εύρος τιμών μίας ιδιότητας αυτού του είδους (datatype property) μπορεί να είναι είτε ένας συγκεκριμένος τύπος δεδομένων (π.χ. ακέραιος), είτε ένα σύνολο από τιμές. Για παράδειγμα, ο χρήστης μπορεί να ορίσει πως η ιδιότητα (datatype property) "έχει_χρονιά" της κλάσης "Έτος_Σοδειάς" (σε μία οντολογία οινολογίας) είναι ένας θετικός ακέραιος αριθμός (π.χ. 1998), ή πως η ίδια ιδιότητα μπορεί να πάρει τιμή "1998" ή "1999" ή "2000".

Οι ιδιότητες στο ODM μπορούν να συσχετίζονται μεταξύ τους. Πιο συγκεκριμένα μία ιδιότητα μπορεί να οριστεί σαν ισοδύναμη μιας άλλης ιδιότητας. Ο ορισμός αυτός πραγματοποιείται μέσω της σχέσης «equivalentProperty_As» όπως φαίνεται στην εικόνα 17. Επιπλέον στο ODM, μία ιδιότητα (είτε Object Property είτε Datatype Property) μπορεί να είναι υπό-ιδιότητα μιας άλλης ιδιότητας, επιτρέποντας με αυτό τον τρόπο τον ορισμό ιεραρχιών από ιδιότητες. Για παράδειγμα, θεωρούμε την ιδιότητα «έχει_χρώμα» η οποία έχει σαν υποκείμενο (domain) την κλάση «Κρασί» και αντικείμενο (range) την κλάση «Χρώμα_Κρασιού» σε μία οντολογία οινολογίας. Η ιδιότητα «έχει_περιγραφή» με το ίδιο υποκείμενο (domain) και αντικείμενο (range) την κλάση «Περιγραφή_Κρασιού» η οποία είναι υπέρ-κλάση της κλάσης «Χρώμα_Κρασιού» μπορεί να οριστεί σαν υπέρ-ιδιότητα της ιδιότητας «έχει_χρώμα».

**Περιορισμοί Ιδιοτήτων (Property Restrictions) στο ODM**

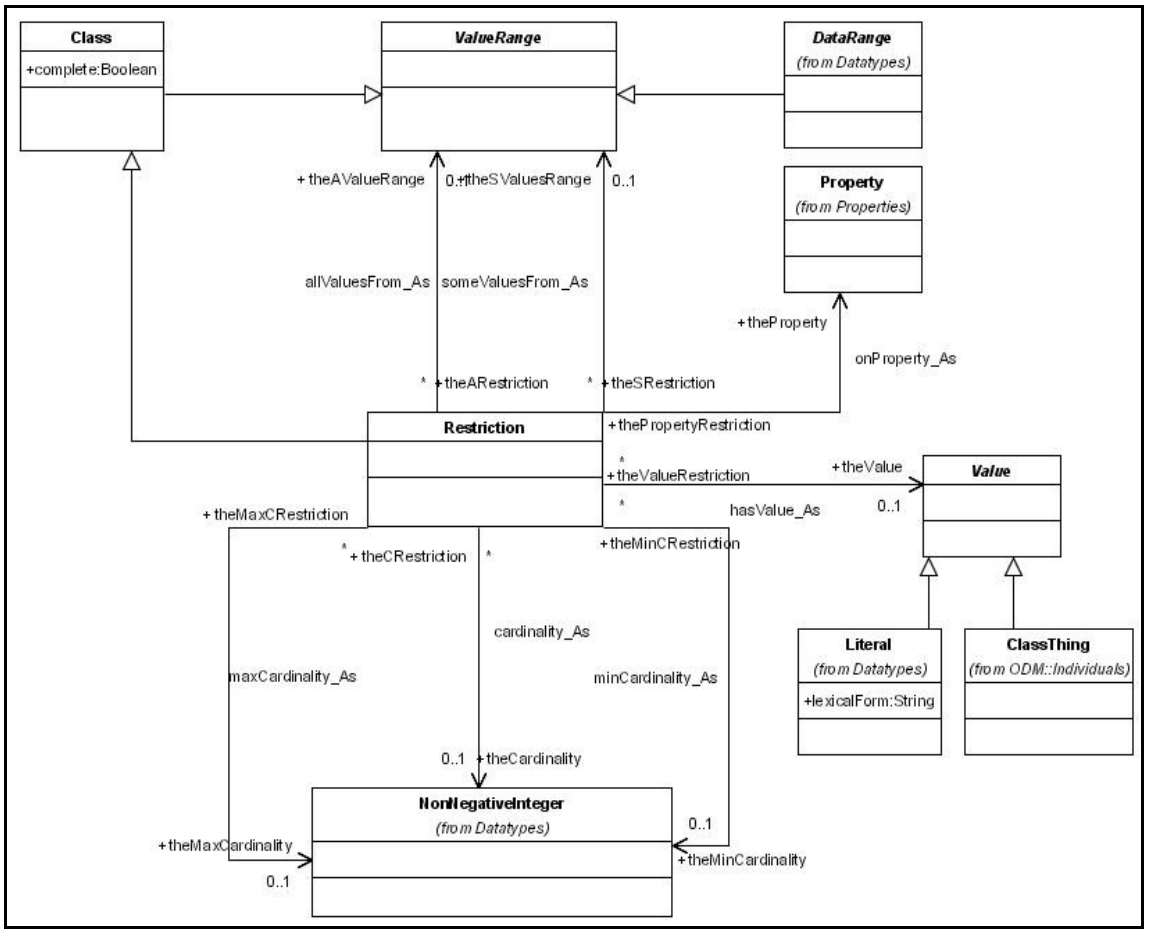
Το ODM ορίζει τον περιορισμό (restriction) σαν μία ειδική κλάση η οποία συνδέεται (μέσω της σχέσης "onProperty_As" που φαίνεται στην εικόνα 18) με μία ιδιότητα η οποία έχει ήδη οριστεί. Πρόκειται για μία ανώνυμη κλάση, δηλαδή μία κλάση που αποτελείται από στιγμιότυπα τα οποία ικανοποιούν κάποιο συγκεκριμένο περιορισμό σε μία ιδιότητα. Υπάρχουν δύο είδη περιορισμών σε ιδιότητες: οι περιορισμοί του αριθμού των στιγμιότυπων των ιδιοτήτων (cardinality constraints) και οι περιορισμοί των τιμών που μπορούν να πάρουν τα στιγμιότυπα των ιδιοτήτων (value constraints).

Κάθε στιγμιότυπο μίας κλάσης μπορεί να έχει έναν αυθαίρετα μεγάλο αριθμό τιμών για κάποια συγκεκριμένη ιδιότητα. Ένας περιορισμός πλήθους (cardinality constraint) θέτει περιορισμούς στον αριθμό των τιμών που μπορεί να πάρει κάποια ιδιότητα, στο πλαίσιο της συγκεκριμένης κλάσης. Για παράδειγμα η κλάση "Κρασί" σε μία οντολογία οινολογίας έχει μία ιδιότητα με όνομα "έχει_χρώμα" η οποία συσχετίζει ένα στιγμιότυπο της κλάσης

"Κρασί" με ένα στιγμιότυπο της κλάσης "Χρώμα_Κρασιού". Στο συγκεκριμένο παράδειγμα, ένας περιορισμός πλήθους (cardinality constraint) πρέπει να ορίζει πως, για την σωστή περιγραφή ενός κρασιού, σε κάθε κρασί θα πρέπει υποχρεωτικά να έχει αντιστοιχηθεί ένα και μοναδικό χρώμα. Κατά συνέπεια η ιδιότητα "έχει_χρώμα" της κλάσης "Κρασί" θα πρέπει να έχει περιορισμό "cardinality=1" για την συγκεκριμένη κλάση. Τυπικά, οι περιορισμοί πλήθους (cardinality constraints) χρησιμοποιούνται για να συνδέσουν έναν περιορισμό με έναν συγκεκριμένο θετικό ακέραιο αριθμό ο οποίος ανήκει στο εύρος τιμών του αντίστοιχου XML Schema τύπου δεδομένων (NonNegativeInteger).

Το ODM ορίζει τρεις τύπους περιορισμών πλήθους (cardinality constraints):

i. Ο περιορισμός ελάχιστου πλήθους (minCardinality): Ο περιορισμός περιγράφει μία κλάση αποτελούμενη από στιγμιότυπα τα οποία έχουν τουλάχιστον N σημασιολογικές διακριτές τιμές όσον αφορά την ιδιότητα πάνω στην οποία έχει εφαρμοστεί ο περιορισμός, όπου N είναι η τιμή του περιορισμού (ένας θετικός ακέραιος αριθμός).
ii. Ο περιορισμός μέγιστου πλήθους (maxCardinality): Ο περιορισμός περιγράφει μία κλάση αποτελούμενη από στιγμιότυπα τα οποία έχουν το πολύ N σημασιολογικές διακριτές τιμές όσον αφορά την ιδιότητα πάνω στην οποία έχει εφαρμοστεί ο περιορισμός, όπου N είναι η τιμή του περιορισμού (ένας θετικός ακέραιος αριθμός).
iii. Ο περιορισμός του πλήθους (cardinality): Ο περιορισμός περιγράφει μία κλάση αποτελούμενη από στιγμιότυπα τα οποία έχουν ακριβώς N σημασιολογικές διακριτές τιμές όσον αφορά την ιδιότητα πάνω στην οποία έχει εφαρμοστεί ο περιορισμός, όπου N είναι η τιμή του περιορισμού (ένας θετικός ακέραιος αριθμός).




**Εικόνα 18: Περιορισμοί Ιδιοτήτων (Property Restrictions) στο ODM**

Ένας περιορισμός τιμών (value constraint) θέτει περιορισμούς στο εύρος τιμών μίας ιδιότητας και εφαρμόζεται στην κλάση στην οποία ορίζεται ο περιορισμός. Προφανώς η κλάση αυτή πρέπει να περιέχει την ιδιότητα (δηλαδή είναι domain της ιδιότητας). Για παράδειγμα θεωρούμε σε μία οντολογία οινολογίας την κλάση «Κρασί». Επιθυμούμε να ορίσουμε πως κάθε στιγμιότυπο της κλάσης «Κρασί» θα πρέπει να ορίζει τουλάχιστον μία από τις περιοχές στις οποίες παρασκευάζεται (βρίσκεται). Αυτό μπορεί να επιτευχθεί μέσω του προσδιορισμού ενός περιορισμού τιμών (value constraint) στην ιδιότητα «βρίσκεται» της οντολογίας η οποία συνδέει την κλάση «Αντικείμενο», η οποία είναι υπέρ-κλάση της κλάσης «Κρασί», με την κλάση «Περιοχή». Ο περιορισμός θα ορίζεται στην κλάση «Κρασί» και θα ορίζει πως ένα στιγμιότυπο της κλάσης «Κρασί» θα πρέπει να δείχνει σε μερικές τιμές (someValuesFrom) από την κλάση «Περιοχή».

Το ODM ορίζει τρεις τύπους περιορισμών τιμών (value constraints):

i. Ο περιορισμός (καθορισμός) όλων των τιμών μίας ιδιότητας (allValuesFrom): Ο περιορισμός συνδέεται είτε με μία κλάση (class), είτε με ένα «DataRange» το οποίο με την σειρά του μπορεί να είναι είτε ένας Τύπος Δεδομένων (Data Type) είτε μία απαρίθμηση (enumeration) τιμών (literals). Η σύνδεση πραγματοποιείται μέσω της αφηρημένης κλάσης «ValueRange» και της σχέσης «allValuesFrom_As», όπως

φαίνεται στην εικόνα 18. Ο περιορισμός χρησιμοποιείται για την περιγραφή μίας κλάσης αποτελούμενη από στιγμιότυπα τα οποία προκύπτουν από τις τιμές της υπό περιορισμό ιδιότητας και τα οποία είναι είτε μέλη της επέκτασης της σχετικής κλάσης (class extension) είτε τιμές (data values) που ανήκουν στο συγκεκριμένο εύρος τιμών που ορίστηκε.

ii. Ο περιορισμός (καθορισμός) μερικών από τις τιμές μίας ιδιότητας (someValuesFrom): Ο περιορισμός συνδέεται είτε με κλάση (class), είτε με ένα «DataRange» το οποίο με την σειρά του μπορεί να είναι είτε ένας Τύπος Δεδομένων (Data Type) είτε μία απαρίθμηση (enumeration) τιμών (literals). Η σύνδεση πραγματοποιείται μέσω της αφηρημένης κλάσης «ValueRange» και της σχέσης «someValuesFrom_As», όπως φαίνεται στην εικόνα 18. Ο περιορισμός χρησιμοποιείται για την περιγραφή μίας κλάσης αποτελούμενη από στιγμιότυπα για τα οποία τουλάχιστον μία τιμή της υπό περιορισμό ιδιότητας είναι είτε στιγμιότυπο της επέκτασης της σχετικής κλάσης (class extension) είτε τιμή (data value) που ανήκει στο συγκεκριμένο εύρος τιμών που ορίστηκε.

iii. Ο περιορισμός (καθορισμός) της τιμής μίας ιδιότητας (hasValue): Ο περιορισμός συνδέεται είτε με ένα στιγμιότυπο είτε με μία τιμή (data value). Η σύνδεση πραγματοποιείται μέσω της αφηρημένης κλάσης «Value» και της σχέσης «hasValue_As», όπως φαίνεται στην εικόνα 18. Περιγράφει έναν περιορισμό για όλα τα individuals για τα οποία η υπό περιορισμό ιδιότητα έχει τουλάχιστον μία τιμή η οποία είναι σημασιολογικά ίση με αυτήν που προσδιορίστηκε.

**Αξιώματα Ιδιοτήτων (Property Axioms) στο ODM**

Σε κάθε ιδιότητα στο ODM δίνεται η δυνατότητα του ορισμού αξιωμάτων (property axioms), τα οποία ορίζουν τα χαρακτηριστικά της ιδιότητας αυτής. Στην πιο απλή του μορφή, ένα αξίωμα απλά ορίζει την ύπαρξη μίας ιδιότητας. Αυτό συμβαίνει όταν η ODM έννοια Ιδιότητα (Property) αποκτά στιγμιότυπα. Το ODM υποστηρίζει τις ακόλουθες δομές υλοποίησης αξιωμάτων των ιδιοτήτων (property axioms):

i. Η δυνατότητα δημιουργίας ιεραρχιών από ιδιότητες μέσω της σχέσης «subPropertyOf_As», όπως φαίνεται στην εικόνα 17. Το αξίωμα ορίζει ότι μία ιδιότητα είναι υπό-ιδιότητα (sub-property) μίας άλλης ιδιότητας.

ii. Το αξίωμα ορισμού των εννοιών «Domain» και «Range» μίας ιδιότητας: Η έννοια «Domain» αντιστοιχεί στην κλάση για την οποία η ιδιότητα έχει οριστεί, ενώ η έννοια «Range» αντιστοιχεί στο εύρος τιμών που μπορεί να πάρει η ιδιότητα. Το εύρος αυτό των τιμών μίας ιδιότητας μπορεί να τεθεί σαν ένας τύπος δεδομένων

(π.χ. String), σαν μία απαρίθμηση (Enumeration) τιμών (Literals) (π.χ. ένα από τα {"Κόκκινο", "Άσπρο", "Κίτρινο"}), ή σαν μία κλάση υπονοώντας ότι ένα στιγμιότυπο της κλάσης αυτής μπορεί να είναι η τιμή της ιδιότητας.

iii. Το αξίωμα ισοδυναμίας (equivalentProperty). Το αξίωμα ορίζει δύο ιδιότητες σαν ισοδύναμες μεταξύ τους.

iv. Το αξίωμα της αντίστροφης ιδιότητας (inverseOf). Το αξίωμα περιγράφει μία ιδιότητα ως αντίστροφη μίας άλλης ιδιότητας. Το Domain της μίας ιδιότητας είναι το Range της άλλης ιδιότητας και αντίστροφα.

v. Γενικοί περιορισμοί πλήθους (global cardinality constraints):

   a. Ο ορισμός μίας ιδιότητας (Object Property ή Datatype Property) ως Functional: Όταν μία ιδιότητα χαρακτηρίζεται ως Functional τότε ισχύει πως η συγκεκριμένη ιδιότητα μπορεί να έχει μόνο μία (μοναδική) τιμή για κάθε στιγμιότυπο.

   b. Ο ορισμός μίας ιδιότητας (Object Property) ως InverseFunctional: Όταν μία ιδιότητα χαρακτηρίζεται ως InverseFunctional τότε ισχύει πως η τιμή της συγκεκριμένης ιδιότητας καθορίζει μοναδικά κάποιο στιγμιότυπο. Τέτοιου είδους ιδιότητες μοιάζουν με την έννοια του κλειδιού στις βάσεις δεδομένων.

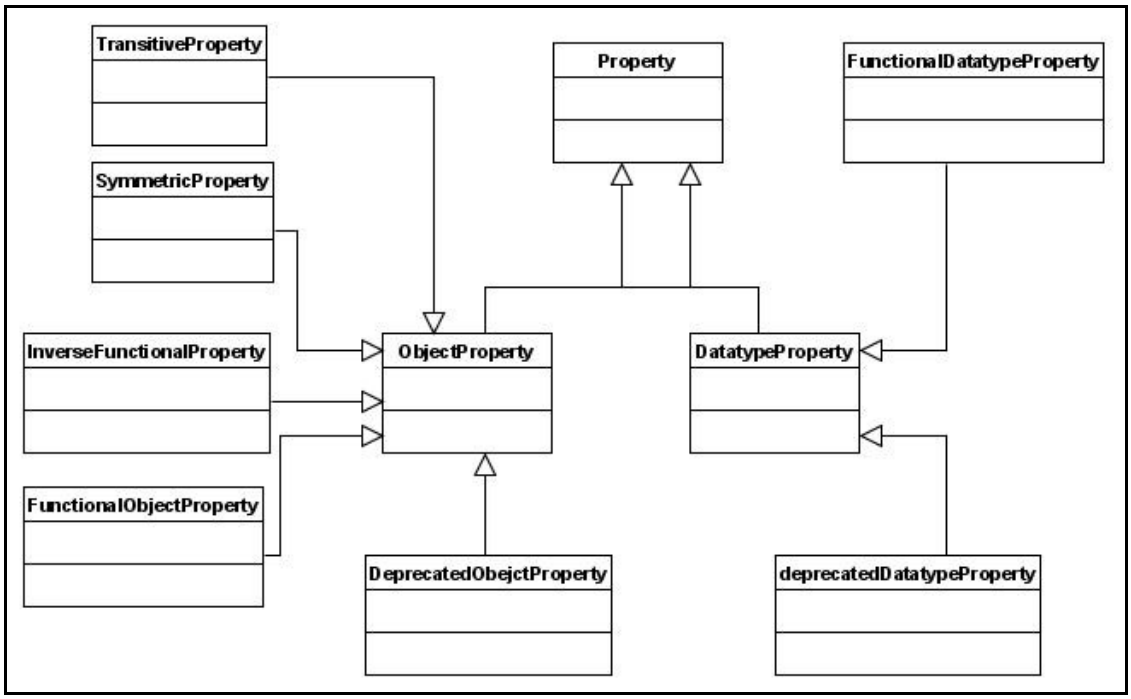
vi. Λογικά χαρακτηριστικά ιδιοτήτων (logical property characteristics):

   a. Ο ορισμός μίας ιδιότητας (Object Property) ως Symmetric: Όταν μία ιδιότητα χαρακτηρίζεται ως Symmetric τότε ισχύει ότι εάν το ζεύγος (x, y) είναι στιγμιότυπο της ιδιότητας, τότε το ζεύγος (y, x) είναι επίσης στιγμιότυπο της ιδιότητας. Μία Symmetric ιδιότητα ορίζεται μέσω της ODM class odm:SymmetricProperty η οποία είναι υπό-κλάση της ODM class odm:ObjectProperty.

   b. Ο ορισμός μίας ιδιότητας (Object Property) ως Transitive: Όταν μία ιδιότητα χαρακτηρίζεται ως Transitive τότε ισχύει πως αν το ζεύγος (x, y) είναι στιγμιότυπο της ιδιότητας, και το ζεύγος (y, z) είναι επίσης στιγμιότυπο της ιδιότητας, τότε το ζεύγος (x, z) είναι επίσης στιγμιότυπο της ιδιότητας. Μία Transitive ιδιότητα ορίζεται μέσω της ODM class odm:TransitiveProperty η οποία είναι υπό-κλάση της ODM class odm:ObjectProperty.

   c. Ο ορισμός μίας ιδιότητας (Object Property ή Datatype Property) ως Deprecated: Όταν μία ιδιότητα χαρακτηρίζεται ως Deprecated τότε ισχύει

πως η ιδιότητα δεν χρησιμοποιείται πλέον και τυπικά έχει αντικατασταθεί από κάποια άλλη ιδιότητα.




**Εικόνα 19: Δομές στο ODM για την αναπαράσταση Αξιωμάτων Ιδιοτήτων (Property Axioms)**

### Στιγμιότυπα (Things ή Individuals) στο ODM

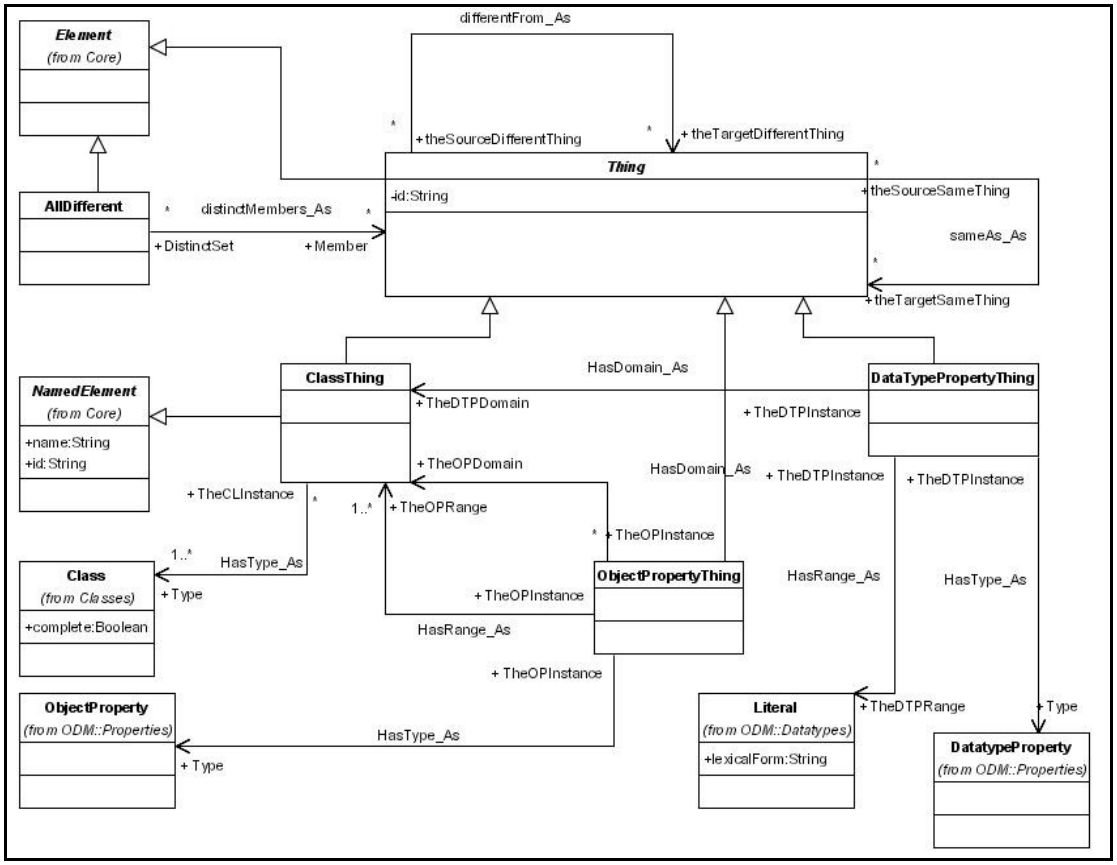
Οι κλάσεις παρέχουν έναν αφαιρετικό μηχανισμό ομαδοποίησης στοιχείων με παρόμοια χαρακτηριστικά. Κάθε κλάση συνδέεται με ένα σύνολο από στιγμιότυπα, το οποίο ονομάζεται επέκταση της κλάσης (class extension). Το στοιχείο «Thing» είναι μία αφηρημένη κλάση που χρησιμοποιείται για την αναπαράσταση όλων των στιγμιότυπων (είτε πρόκειται για στιγμιότυπα κλάσεων, είτε για στιγμιότυπα ιδιοτήτων) σε μία οντολογία. Επιπλέον, ένα στιγμιότυπο μπορεί να είναι διαφορετικό (ορίζεται μέσω της σχέσης «differentFrom_As» όπως φαίνεται στην εικόνα 20) από κάποιο άλλο στιγμιότυπο. Για παράδειγμα το στιγμιότυπο «Γλυκό» της κλάσης «Ζάχαρι_Κρασιού» σε μία οντολογία οινολογίας είναι διαφορετικό από το στιγμιότυπο «Ξηρό» της ίδιας κλάσης. Ακόμα ένα στιγμιότυπο μπορεί να είναι ίδιο (ορίζεται μέσω της σχέσης «sameAs_As» όπως φαίνεται στην εικόνα 20) με κάποιο άλλο στιγμιότυπο. Για παράδειγμα το στιγμιότυπο «Εργαστήριο_MUSIC» της κλάσης «Εργαστήρια_Πολυτεχνείου_Κρήτης» (μίας οντολογίας περιγραφής του Πολυτεχνείου Κρήτης) είναι το ίδιο με το στιγμιότυπο «MUSIC» της κλάσης «Μέλη_DBE» (μίας οντολογίας περιγραφής του προγράμματος DBE).

Το ODM ορίζει τρεις υπό-κλάσεις της ODM class «Thing», δηλαδή ουσιαστικά ορίζει τριών ειδών στιγμιότυπα:

i. Τα στιγμιότυπα κλάσεων (ClassThings): Η δομή χρησιμοποιείται για την αναπαράσταση των στιγμιότυπων μιας κλάσης. Κάθε στιγμιότυπο μίας κλάσης θα πρέπει να ορίζει υποχρεωτικά τον τύπο του (κλάση).

ii. Τα στιγμιότυπα ιδιοτήτων τύπου «Object Property» (ObjectPropertyThings): Η δομή αναπαριστά τα στιγμιότυπα των ιδιοτήτων (Object Properties) μίας κλάσης. Αυτό σημαίνει πως συσχετίζει στιγμιότυπα κλάσεων (ClassThings) με άλλα στιγμιότυπα κλάσεων (ClassThings). Κάθε «ObjectPropertyThing» θα πρέπει να ορίζει υποχρεωτικά τον τύπο του (ιδιότητα τύπου «Object Property»).

iii. Τα στιγμιότυπα ιδιοτήτων τύπου «Datatype Property» (DatatypePropertyThings): Η δομή αναπαριστά τα στιγμιότυπα των ιδιοτήτων (Datatype Properties) μίας κλάσης. Αυτό σημαίνει πως συσχετίζει ένα στιγμιότυπο κλάσης (ClassThing) με μία τιμή (Literal: data value). Κάθε «DataTypePropertyThing» θα πρέπει να ορίζει υποχρεωτικά τον τύπο του (ιδιότητα τύπου «Datatype Property»).




**Εικόνα 20: Στιγμιότυπα (Individuals) στο ODM**

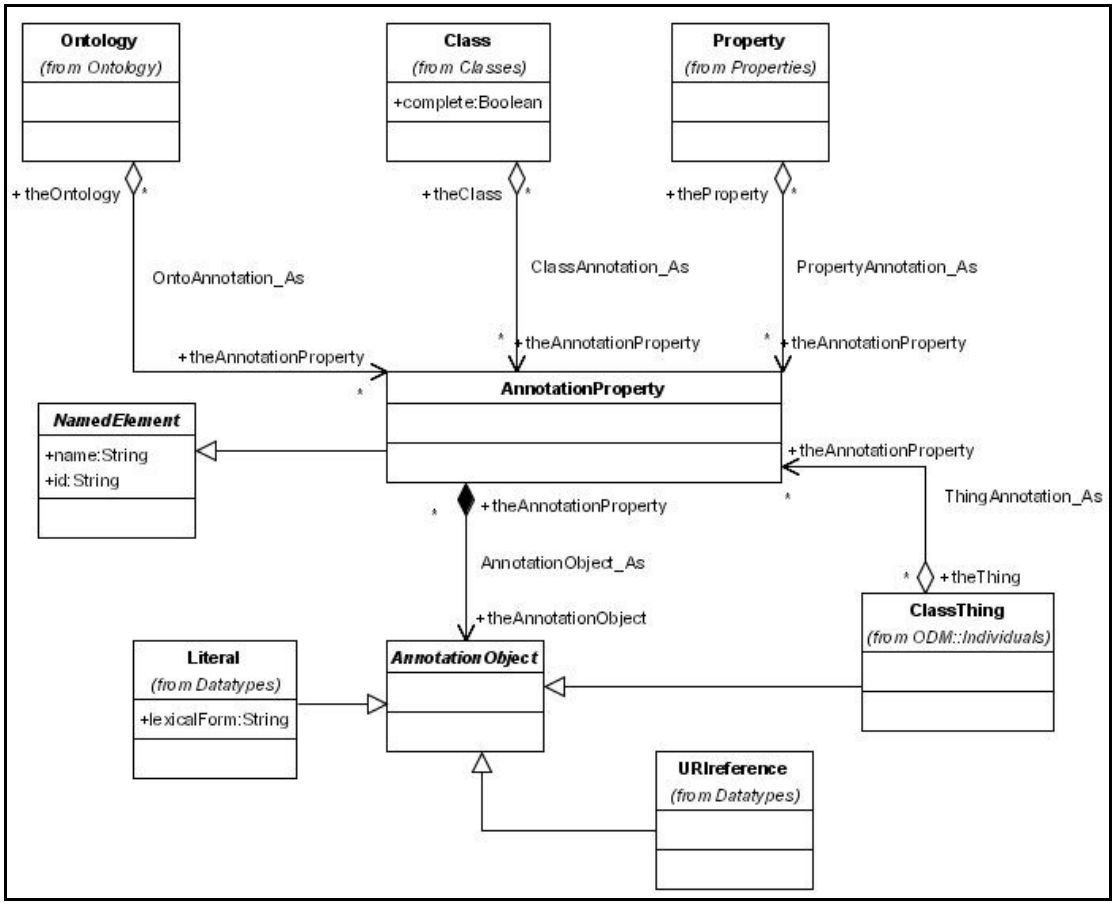
### Ορισμός Διακριτών Συνόλων από Στιγμιότυπα (Individuals)

Όταν δημιουργεί μία οντολογία, ο χρήστης μπορεί να θελήσει να ορίσει ότι κάποια στιγμιότυπα (individuals) σε μία λίστα είναι όλα μεταξύ τους διαφορετικά. Αντί να δηλώσει για κάθε στιγμιότυπο πως είναι διαφορετικό από τα υπόλοιπα, μπορεί να χρησιμοποιήσει

την δομή «AllDifferent» για να ορίσει ότι όλα τα μέλη αυτού του συνόλου είναι διαφορετικά μεταξύ τους.

### Σχόλια (Annotations) στο ODM

Πολλές φορές είναι συνηθισμένο για τον δημιουργό μίας οντολογίας να επιθυμεί να σχολιάσει κάποιες από τις έννοιες που ορίζει. Για τον σκοπό αυτό το ODM εισάγει ένα σύνολο από προκαθορισμένες έννοιες οι οποίες επιτρέπουν στον δημιουργό μίας οντολογίας να σχολιάσει της έννοιες που επιθυμεί. Η εικόνα 21 δείχνει το τμήμα αυτό του μετά-μοντέλου.




**Εικόνα 21: Σχόλια (Annotations) στο ODM**

Το ODM επιτρέπει σχόλια πάνω σε κλάσεις, σε ιδιότητες, σε στιγμιότυπα κλάσεων (ClassThings) και σε οντολογίες χρησιμοποιώντας την έννοια «AnnotationProperty». Η έννοια «AnnotationProperty» συνδέεται με τις προαναφερόμενες έννοιες (κλάσεις, ιδιότητες κτλ.) μέσω των σχέσεων «ClassAnnotation_As», «PropertyAnnotation_As», «ThingAnnotation_As» και «OntoAnnotation_As» αντίστοιχα. Η έννοια «AnnotationObject» ενός «AnnotationProperty» (δηλαδή η «τιμή» του σχολίου) πρέπει να είναι είτε μία τιμή (Literal: data value), είτε μία URI παραπομπή (URIreference), είτε ένα

στιγμιότυπο μίας κλάσης. Η OWL DL προκαθορίζει κάποιες συγκεκριμένες δομές σχολιασμού όπως τα comment, label, versionInfo, seeAlso και isDefinedBy. Το ODM επιλέγει να μην προκαθορίσει συγκεκριμένες δομές σχολιασμού. Αντί για αυτό παρέχει τον μηχανισμό για τον προσδιορισμό σχολίων όπως φαίνεται στην εικόνα 21. Τα εργαλεία τα οποία θα χρησιμοποιηθούν για την μοντελοποίηση με το ODM μπορεί να επιλέξουν να εισάγουν τις προκαθορισμένες OWL δομές σχολιασμού. Στο εργαλείο που δημιουργήθηκε για τις ανάγκες της παρούσας διπλωματικής διατριβής αποφασίστηκε ο προκαθορισμός και η χρησιμοποίηση των εξής δομών σχολιασμού: Comment, Label, versionInfo, seeAlso, και isDefinedBy.

**Τύποι Δεδομένων (Data Types) στο ODM**

Η έννοια του Εύρους Τιμών (Data Range) έχει αναφερθεί επανειλημμένα μέχρι τώρα με σκοπό τον προσδιορισμό του εύρους τιμών τις οποίες μπορεί να πάρει μία ιδιότητα. Το ODM ορίζει δύο είδη προσδιορισμών του Εύρους Τιμών:

i. Ο προσδιορισμός των Τύπων Δεδομένων (Datatype). Προσδιορίζει ένα πλαίσιο ορισμού και χρήσης τύπων δεδομένων το οποίο είναι κατάλληλο για την αναφορά στους XML Schema τύπους δεδομένων. Πιο συγκεκριμένα το ODM ορίζει την έννοια «DataType» η οποία θα χρησιμοποιηθεί για την αναπαράσταση όλων των τύπων δεδομένων που ορίζονται στο M1 επίπεδο (σύμφωνα με το MOF πρότυπο). Κάθε τύπος δεδομένων πρέπει να παρέχει μία παραπομπή (URI) στον ορισμό του. Οι XML Schema τύποι δεδομένων αναμένονται να ορίζονται μία φορά από τα ODM εργαλεία μοντελοποίησης και να εισάγονται σε οντολογίες όταν γίνεται η μοντελοποίηση. Για παράδειγμα, ο τύπος δεδομένων XML Schema String πρέπει να οριστεί μέσω της έννοιας «DataType» με URI παραπομπή (URIreference) http://www.w3.org/2001/XMLSchema#string το οποίο καθορίζει την τοποθεσία ορισμού του τύπου. Οι τιμές (data values) στο ODM είναι στιγμιότυπα της ODM κλάσης «Literal». Οι τιμές αυτές (literals) μπορούν να είναι απλές (plain) ή «ορισμένες» σε κάποιο URI (typed). Οι «ορισμένες» τιμές (typed literals) περιέχουν μία URI παραπομπή (URIreference) που υποδηλώνει την τοποθεσία ορισμού του τύπου τους.

ii. Ο προσδιορισμός των Απαριθμημένων Τιμών (Enumeration). Επιπρόσθετα με τους XML Schema τύπους δεδομένων, το ODM παρέχει μία επιπλέον δομή για τον προσδιορισμό ενός εύρους τιμών, που ονομάζεται «Enumeration». Η δομή αυτή αποτελεί ένα πλαίσιο (container) το οποίο περιέχει ένα διατεταγμένο σύνολο από τιμές (Literals).

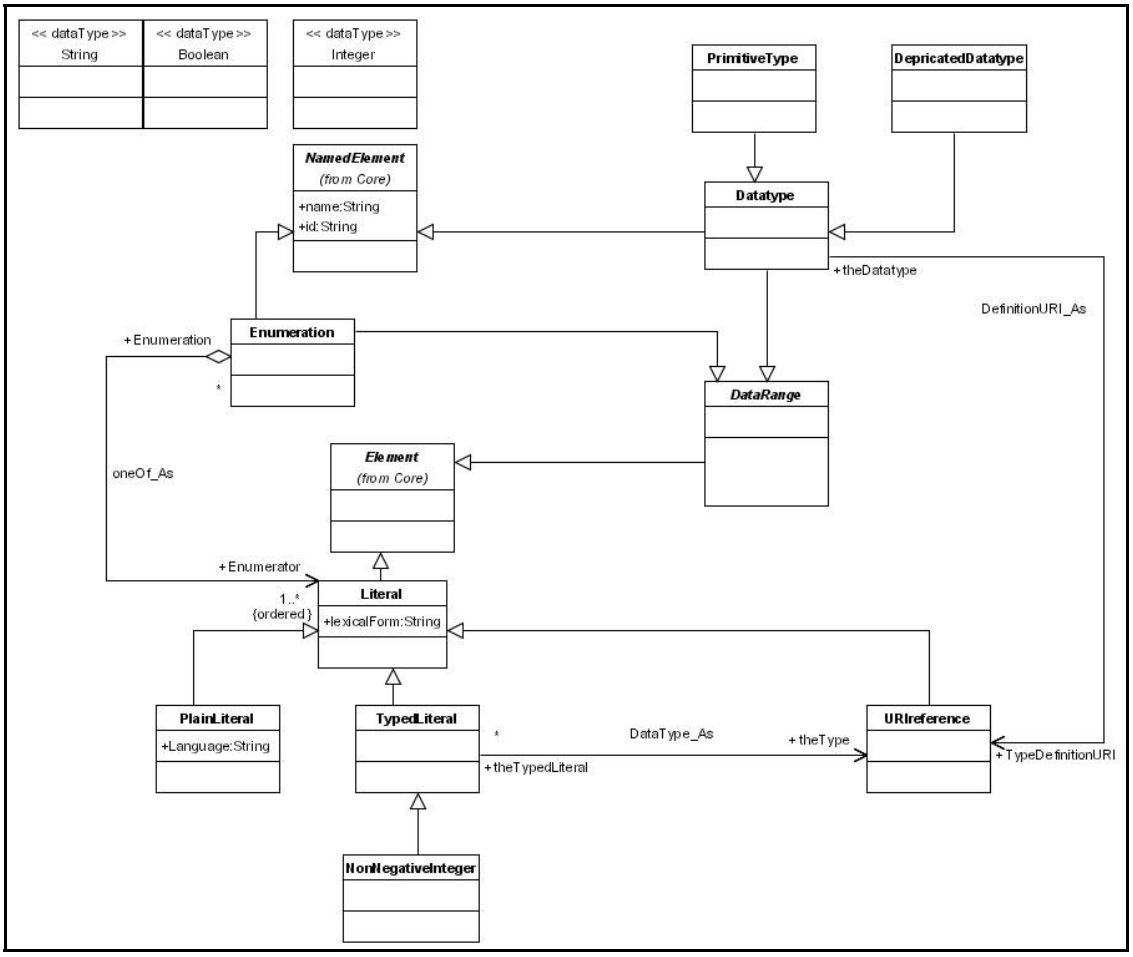
Οι XML Schema τύποι δεδομένων οι οποίοι αναμένονται να χρησιμοποιηθούν κατά την διάρκεια μοντελοποίησης με το ODM είναι οι παρακάτω:

Ο βασικός (primitive) τύπος δεδομένων xsd:string, και επιπλέον οι τύποι δεδομένων που απορρέουν από το xsd:string: xsd:normalizedString, xsd:token, xsd:language, xsd:NMTOKEN, xsd:Name, και xsd:NCName.

Ο βασικός (primitive) τύπος δεδομένων xsd:boolean. Οι βασικοί (primitive) αριθμητικοί τύποι δεδομένων xsd:decimal, xsd:float, και xsd:double, και επιπλέον όλοι οι τύποι δεδομένων που απορρέουν από το xsd:decimal (xsd:integer, xsd:positiveInteger, xsd:nonPositiveInteger, xsd:negativeInteger, xsd:nonNegativeInteger, xsd:long, xsd:int, xsd:short, xsd:byte, xsd:unsignedLong, xsd:unsignedInt, xsd:unsignedShort, xsd:unsignedByte).

Ο σχετικοί με τον χρόνο βασικοί (primitive) τύποι δεδομένων: xsd:dateTime, xsd:time, xsd:date, xsd:gYearMonth, xsd:gYear, xsd:gMonthDay, xsd:gDay, και xsd:gMonth.

Οι βασικοί (primitive) τύποι δεδομένων xsd:hexBinary, xsd:base64Binary, και xsd:anyURI.




**Εικόνα 22: Τύποι Δεδομένων (Data Types) στο ODM**

**Ανακεφαλαίωση**

Στην ενότητα αυτή περιγράφηκε η αρχιτεκτονική μετά-δεδομένων MOF σύμφωνα με την οποία υλοποιήθηκε το Μετά-μοντέλο Ορισμού Οντολογιών (Ontology Definition Metamodel - ODM) και στη συνέχεια περιγράφηκε το Μετά-μοντέλο Ορισμού Οντολογιών το οποίο αναπτύχθηκε στο πλαίσιο του προγράμματος DBE. Το μετά-μοντέλο αυτό είναι συμβατό με τη γλώσσα OWL(-DL) η οποία είναι η πιο ευρέως αποδεκτή γλώσσα περιγραφής οντολογιών στην κοινότητα του σημασιολογικού ιστού (semantic web). Το γεγονός αυτό επιτρέπει στο μετά-μοντέλο ODM να περιγράφει υπάρχουσες οντολογίες οι οποίες έχουν περιγραφεί με τη γλώσσα OWL. Επίσης, η αντίστροφη διαδικασία είναι πιθανή. Δηλαδή, οντολογίες που έχουν περιγραφεί με το ODM μπορούν επίσης να περιγραφούν σε OWL. Γεγονός το οποίο σημαίνει ότι ο γραφικός επεξεργαστής που υλοποιήθηκε στο πλαίσιο αυτής της διπλωματικής μπορεί με μικρές επεκτάσεις να χρησιμοποιηθεί για την δημιουργία OWL οντολογιών.

Κεφάλαιο 5

# ΠΕΡΙΠΤΩΣΕΙΣ ΧΡΗΣΗΣ (USE CASES)

**Εισαγωγή**

Χρησιμοποιώντας το εργαλείο που υλοποιήθηκε στο πλαίσιο της παρούσας διπλωματικής διατριβής (GRAMOFONE), ο χρήστης μπορεί να δημιουργήσει ODM (Ontology Definition Metamodel) μοντέλα τα οποία αποτελούν στιγμιότυπα (instances) του αντίστοιχου ODM μετά-μοντέλου. Ουσιαστικά λοιπόν μέσω του GRAMOFONE προσδιορίζονται μοντέλα επιπέδου M1 (όπως ορίζεται από το MOF πρότυπο το οποίο περιγράφηκε στο κεφάλαιο 4) για την περιγραφή οντολογιών.

Οι λειτουργικές πτυχές του GRAMOFONE περιγράφονται μέσω της μεθοδολογίας των περιπτώσεων χρήσης (Use Cases). Οι περιπτώσεις χρήσης είναι μία software-engineering τεχνολογία η οποία παρέχει μία τυπική (formal) περιγραφή της λειτουργικότητας ενός συστήματος, η οποία είναι πλήρως κατανοητή σε οποιονδήποτε επιθυμήσει να την διαβάσει. Η φόρμα (template) περιγραφής των περιπτώσεων χρήσης του GRAMOFONE η οποία χρησιμοποιείται σε αυτό το κεφάλαιο είναι η ευρέως γνωστή φόρμα περιγραφής περιπτώσεων χρήσης που ορίζεται από τον Alistair Cockburn στο βιβλίο του «Writing Effective Use Cases». [21]

Μία περίπτωση χρήσης αποτελεί ένα είδος «συμβολαίου» (συμφωνίας) μεταξύ των χρηστών ενός συστήματος όσον αφορά την συμπεριφορά του συστήματος αυτού. Μία περίπτωση χρήσης περιγράφει την συμπεριφορά του συστήματος κάτω από διάφορες συνθήκες καθώς το σύστημα ανταποκρίνεται σε μία αίτηση (request) από έναν από τους χρήστες, ο οποίος ονομάζεται «βασικός χαρακτήρας» (primary actor) της συγκεκριμένης περίπτωσης χρήσης. Ο βασικός χαρακτήρας ξεκινά μία αλληλεπίδραση με το σύστημα με σκοπό την επίτευξη κάποιου στόχου. Το σύστημα ανταποκρίνεται, προστατεύοντας παράλληλα τα συμφέροντα των υπολοίπων χρηστών. Είναι δυνατή η ανάπτυξη διαφορετικών ακολουθιών από συμπεριφορές, ή σενάρια, ανάλογα με τις συγκεκριμένες αιτήσεις που πραγματοποιούνται και τις συνθήκες που επικρατούν κατά την διάρκεια των αιτήσεων αυτών. Η περίπτωση χρήσης συγκεντρώνει όλα αυτά τα διαφορετικά σενάρια μαζί. Οι περιπτώσεις χρήσης

περιγράφονται σε μορφή κειμένου. Αποτελούν μέσο επικοινωνίας μεταξύ των ατόμων, οι οποίοι πολλές φορές δεν έχουν καμία ιδιαίτερη εκπαίδευση. [28]

Οι τρεις βασικές έννοιες που χρησιμοποιούνται στην περιγραφή μίας περίπτωσης χρήσης είναι οι παρακάτω:

- Εμβέλεια (Scope): Περιγράφει το σύστημα στο οποίο λαμβάνει χώρα η περίπτωση χρήσης.
- Βασικός χαρακτήρας (Primary actor): Περιγράφει τον χρήστη ο οποίος σκοπεύει να εκπληρώσει κάποια ενέργεια μέσω της περίπτωσης χρήσης.
- Επίπεδο (Level): Περιγράφει πόσο υψηλού ή χαμηλού επιπέδου είναι ο στόχος της περίπτωσης χρήσης.

Οι όροι που είναι απαραίτητοι για την κατανόηση των περιπτώσεων χρήσης είναι οι παρακάτω:

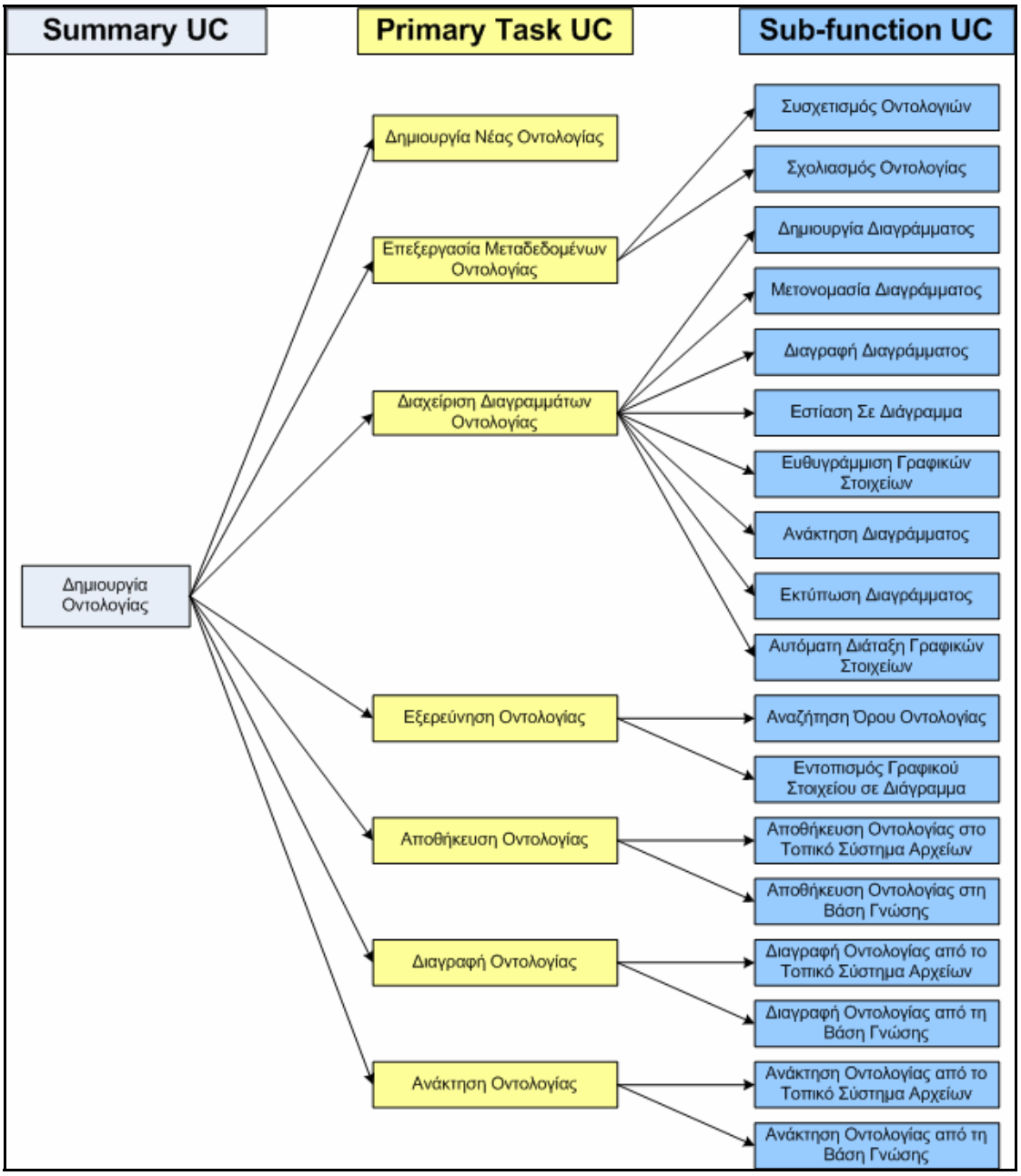
- Στόχος (Goal in Context): Ο στόχος που επιδιώκει ο χρήστης να επιτύχει μέσω της επιτυχημένης πραγματοποίησης της Περίπτωσης Χρήσης.
- Κατάσταση Επιτυχούς Τερματισμού (Success End Condition): Περιγράφει την κατάσταση στην οποία θα βρίσκεται το σύστημα (στο οποίο λαμβάνει χώρα η Περίπτωση Χρήσης) στην περίπτωση που ακολουθηθεί το επιτυχημένο σενάριο (main success scenario) της Περίπτωσης Χρήσης.
- Κατάσταση Ανεπιτυχούς Τερματισμού (Failed End Condition): Περιγράφει την κατάσταση στην οποία θα βρίσκεται το σύστημα (στο οποίο λαμβάνει χώρα η Περίπτωση Χρήσης) στην περίπτωση που δεν ακολουθηθεί το επιτυχημένο σενάριο (main success scenario) της Περίπτωσης Χρήσης (δηλαδή αποτύχει).
- Βασικός χαρακτήρας (Primary actor): ο χρήστης ο οποίος ξεκινά μία αλληλεπίδραση με το σύστημα για την επίτευξη κάποιου στόχου.
- Δευτερεύων χαρακτήρας (Secondary actor): Ο χρήστης ο οποίος συμμετέχει στην περίπτωση χρήσης, ωστόσο δεν είναι εκείνος ο οποίος την έχει ξεκινήσει (αυτός είναι ο βασικός χαρακτήρας).
- Περίπτωση χρήσης (Use case): ένα είδος «συμβολαίου» (συμφωνίας) της συμπεριφοράς του υπό εξέταση συστήματος.
- Πεδίο δράσης (Scope): υποδεικνύει το σύστημα που εξετάζεται.
- Προϋποθέσεις και εγγυήσεις (Precondition and guarantees): οι συνθήκες που πρέπει να ισχύουν πριν και μετά την εκτέλεση της περίπτωσης χρήσης.
- Έναυσμα (Trigger): η ενέργεια που προηγείται και ενεργοποιεί την περίπτωση χρήσης.

- Επιτυχημένο σενάριο (Main success scenario): η περίπτωση στην οποία δεν θα υπάρξουν καθόλου σφάλματα.
- Επεκτάσεις (Extensions): οι διαφορετικές εκδοχές της πορείας του σεναρίου. Χρησιμοποιούνται για την περιγραφή καταστάσεων που προκύπτουν από σφάλματα στην πορεία του σεναρίου της περίπτωσης χρήσης. Οι αριθμοί των επεκτάσεων αναφέρονται στα βήματα του επιτυχημένου σεναρίου στα οποία αναγνωρίζεται κάθε διαφορετική κατάσταση (π.χ. Τα βήματα 3α και 3β υποδηλώνουν δύο διαφορετικές συνθήκες οι οποίες μπορούν να εμφανιστούν στο βήμα 3).
- Αποκλίσεις (Sub-Variations): οι διαφοροποιήσεις της πορείας του σεναρίου. Χρησιμοποιούνται για την περιγραφή καταστάσεων που προκύπτουν όταν ο υπάρχουν παραπάνω από μία δυνατές εκδοχές εκτέλεσης του σεναρίου της περίπτωσης χρήσης. Οι αριθμοί των επεκτάσεων αναφέρονται στα βήματα του επιτυχημένου σεναρίου στα οποία αναγνωρίζεται κάθε διαφορετική κατάσταση (π.χ. Τα βήματα 3α και 3β υποδηλώνουν δύο διαφορετικές συνθήκες οι οποίες μπορούν να εμφανιστούν στο βήμα 3).
- Όταν μία περίπτωση χρήσης αναφέρεται σε μία άλλη περίπτωση χρήσης, τότε η δεύτερη υπογραμμίζεται.

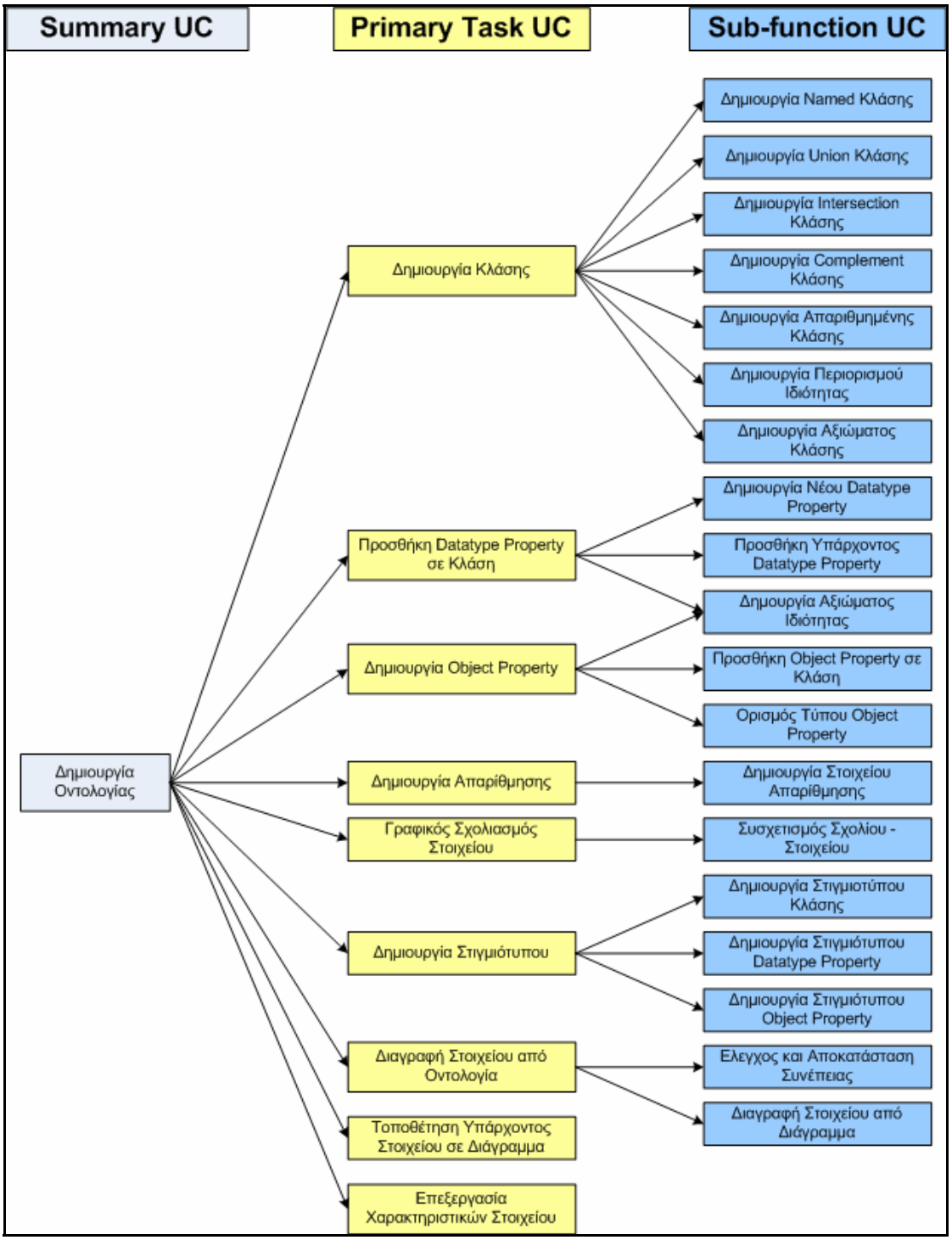
Για να δηλώσουμε πως ο στόχος προς εκπλήρωση μίας περίπτωσης χρήσης έχει δική του υπόσταση, δηλαδή υπάρχει περίπτωση ο χρήστης να χρησιμοποιήσει το σύστημα με μοναδικό σκοπό την πραγματοποίηση του συγκεκριμένου στόχου, ταξινομούμε την περίπτωση χρήσης ως «βασική λειτουργία» (primary task ή user-task level). Παράδειγμα τέτοιας περίπτωσης χρήσης είναι η εγγραφή ενός καινούργιου φοιτητή για ένα σύστημα διαχείρισης των μελών ενός πανεπιστημίου. Η ομαδοποίηση πολλών βασικών λειτουργιών (primary tasks) με σκοπό την επίτευξη κάποιου πιο πολύπλοκου στόχου επιτυγχάνεται με την ταξινόμηση της περίπτωσης χρήσης ως «περίληψη» (summary level ή strategic). Το «summary level» είναι ένα επίπεδο υψηλότερο από το «primary task» επίπεδο. Παράδειγμα τέτοιας περίπτωσης χρήσης είναι η δημιουργία μίας οντολογίας με την χρήση του GRAMOFONE, καθώς για την εκπλήρωσή της απαιτείται η εκπλήρωση πολλών βασικών λειτουργιών (primary tasks). Για τον προσδιορισμό υπό-στόχων οι οποίοι πραγματοποιούνται για την επίτευξη μίας βασικής ενέργειας (primary task), χρησιμοποιείται ο όρος «υπό-λειτουργία» (sub-function level) με τον οποίο μπορεί να χαρακτηριστεί μία περίπτωση χρήσης. Μία βασική λειτουργία μπορεί να αποτελείται από πολλές υπό-λειτουργίες. Το «sub-function level» είναι ένα επίπεδο χαμηλότερο από το «primary task» επίπεδο. Παράδειγμα τέτοιας περίπτωσης χρήσης είναι η είσοδος (log-in) σε ένα σύστημα δανειστικής βιβλιοθήκης.

Με σκοπό την παρουσίαση της γενικής εικόνας του συνόλου των περιπτώσεων χρήσης που θα περιγραφούν παρακάτω θα χρησιμοποιηθούν δύο συμπληρωματικές αναπαραστάσεις:

1. Ένα διάγραμμα το οποίο δείχνει τις περιπτώσεις χρήσης ως κουτιά με συνδέσεις που αναπαριστούν τις μεταξύ τους συσχετίσεις (τις περιπτώσεις χρήσης που ακολουθούν μετά από κάθε περίπτωση χρήσης). Λόγω του μεγάλου αριθμού των περιπτώσεων χρήσης αναγκαστήκαμε να χωρίσουμε το διάγραμμα σε δύο υπό-διαγράμματα.
2. Ένας πίνακας ο οποίος παρουσιάζει τις περιπτώσεις χρήσης περιέχοντας πληροφορίες κάθε περίπτωσης χρήσης οι οποίες περιλαμβάνουν το αναγνωριστικό (id), το επίπεδο (level), το βασικό χαρακτήρα (primary actor), το στόχο (goal) και μία σύντομη περιγραφή κάθε περίπτωσης χρήσης.




**Εικόνα 23: Διάγραμμα Περιπτώσεων Χρήσης Μέρος Α'**




**Εικόνα 24: Διάγραμμα Περιπτώσεων Χρήσης Μέρος Β'**

Ο παρακάτω πίνακας παρουσιάζει τις περιπτώσεις χρήσης ταξινομημένες με βάση το αναγνωριστικό τους (id). Ο πίνακας περιέχει πληροφορίες για κάθε περίπτωση χρήσης. Οι πληροφορίες αυτές περιλαμβάνουν το αναγνωριστικό (id) κάθε περίπτωσης χρήσης, το επίπεδο (level), το βασικό χαρακτήρα (primary actor), το στόχο (goal) και μία σύντομη περιγραφή της περίπτωσης χρήσης.

| # | Level | Primary Actor | Goal | Brief |
|---|---|---|---|---|
| UC_1 | Summary | Χρήστης | Δημιουργία Οντολογίας | Ο χρήστης επιθυμεί να δημιουργήσει ένα ODM Μοντέλο, δηλαδή μία οντολογία |
| UC_2 | Primary Task | Χρήστης | Δημιουργία Νέας Οντολογίας | Ο χρήστης επιθυμεί να δημιουργήσει μία καινούργια (δική του) οντολογία |
| UC_3 | Primary Task | Χρήστης | Επεξεργασία Μεταδεδομένων Οντολογίας | Ο χρήστης επιθυμεί να τροποποιήσει τα χαρακτηριστικά της οντολογίας που επεξεργάζεται όπως αυτά ορίζονται από το ODM |
| UC_4 | Sub-function | Χρήστης | Συσχετισμός Οντολογιών | Ο χρήστης επιθυμεί να συσχετίσει την οντολογία που επεξεργάζεται με άλλες οντολογίες |
| UC_5 | Sub-function | Χρήστης | Σχολιασμός Οντολογίας | Ο χρήστης επιθυμεί να προσθέσει κάποιο σχόλιο στην οντολογία που επεξεργάζεται |
| UC_6 | Primary Task | Χρήστης | Διαχείριση Διαγραμμάτων Οντολογίας | Ο χρήστης επιθυμεί να διαχειριστεί κάποιο από τα διαγράμματα της οντολογίας |
| UC_7 | Sub-function | Χρήστης | Δημιουργία Διαγράμματος | Ο χρήστης επιθυμεί να δημιουργήσει ένα καινούργιο διάγραμμα στο πλαίσιο επεξεργασίας μίας οντολογίας |
| UC_8 | Sub-function | Χρήστης | Μετονομασία Διαγράμματος | Ο χρήστης επιθυμεί να μετονομάσει κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται |
| UC-9 | Sub-function | Χρήστης | Διαγραφή Διαγράμματος | Ο χρήστης επιθυμεί να διαγράψει κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται |
| UC_10 | Sub-function | Χρήστης | Εστίαση Σε Διάγραμμα | Ο χρήστης επιθυμεί να χρησιμοποιήσει την λειτουργία της εστίασης σε ένα διάγραμμα |
| UC_11 | Sub-function | Χρήστης | Ευθυγράμμιση Γραφικών Στοιχείων | Ο χρήστης επιθυμεί να ευθυγραμμίσει κάποια από τα στοιχεία ενός διαγράμματος |
| UC_12 | Sub-function | Χρήστης | Ανάκτηση Διαγράμματος | Ο χρήστης επιθυμεί να ανακτήσει στην επιφάνεια εργασίας κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται |
| UC_13 | Sub-function | Χρήστης | Εκτύπωση Διαγράμματος | Ο χρήστης επιθυμεί να εκτυπώσει κάποιο διάγραμμα |
| UC_14 | Sub-function | Χρήστης | Αυτόματη Διάταξη Γραφικών Στοιχείων | Ο χρήστης επιθυμεί να τοποθετηθούν αυτόματα από το GRAMOFONE τα στοιχεία ενός διαγράμματος |
| UC_15 | Primary Task | Χρήστης | Εξερεύνηση Οντολογίας | Ο χρήστης επιθυμεί να εξερευνήσει τα περιεχόμενα της οντολογίας που επεξεργάζεται |
| UC_16 | Sub-function | Χρήστης | Αναζήτηση Όρου Οντολογίας | Ο χρήστης επιθυμεί να αναζητήσει κάποιον από τους όρους που έχει ορίσει μέσα στην οντολογία που επεξεργάζεται |

| | | | | |
|---|---|---|---|---|
| UC_17 | Sub-function | Χρήστης | Εντοπισμός Γραφικού Στοιχείου σε Διάγραμμα | Ο χρήστης επιθυμεί να εντοπίσει κάποιον όρο της οντολογίας που επεξεργάζεται μέσα σε ένα διάγραμμα |
| UC_18 | Primary Task | Χρήστης | Αποθήκευση Οντολογίας | Ο χρήστης επιθυμεί να αποθηκεύσει την οντολογία που επεξεργάζεται |
| UC_19 | Sub-function | Χρήστης | Αποθήκευση Οντολογίας στο Τοπικό Σύστημα Αρχείων | Ο χρήστης επιθυμεί να αποθηκεύσει την οντολογία που επεξεργάζεται στο τοπικό σύστημα αρχείων |
| UC_20 | Sub-function | Χρήστης | Αποθήκευση Οντολογίας στη Βάση Γνώσης | Ο χρήστης επιθυμεί να αποθηκεύσει την οντολογία που επεξεργάζεται στη Βάση Γνώσης |
| UC_21 | Primary Task | Χρήστης | Διαγραφή Οντολογίας | Ο χρήστης επιθυμεί να διαγράψει μία ή περισσότερες οντολογίες |
| UC_22 | Sub-function | Χρήστης | Διαγραφή Οντολογίας από το Τοπικό Σύστημα Αρχείων | Ο χρήστης επιθυμεί να διαγράψει μία ή περισσότερες οντολογίες από το τοπικό σύστημα αρχείων |
| UC_23 | Sub-function | Χρήστης | Διαγραφή Οντολογίας από τη Βάση Γνώσης | Ο χρήστης επιθυμεί να διαγράψει μία ή περισσότερες οντολογίες από τη Βάση Γνώσης |
| UC_24 | Primary Task | Χρήστης | Ανάκτηση Οντολογίας | Ο χρήστης επιθυμεί να ανακτήσει κάποια υπάρχουσα οντολογία |
| UC_25 | Sub-function | Χρήστης | Ανάκτηση Οντολογίας από το Τοπικό Σύστημα Αρχείων | Ο χρήστης επιθυμεί να ανακτήσει μία υπάρχουσα οντολογία από το τοπικό σύστημα αρχείων |
| UC_26 | Sub-function | Χρήστης | Ανάκτηση Οντολογίας από τη Βάση Γνώσης | Ο χρήστης επιθυμεί να ανακτήσει μία υπάρχουσα οντολογία από τη Βάση Γνώσης |
| UC_27 | Primary Task | Χρήστης | Δημιουργία Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει μία Κλάση |
| UC_28 | Sub-function | Χρήστης | Δημιουργία Named Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει μία Named Κλάση |
| UC_29 | Sub-function | Χρήστης | Δημιουργία Union Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει μία Union Κλάση |
| UC_30 | Sub-function | Χρήστης | Δημιουργία Intersection Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει μία Intersection Κλάση |
| UC_31 | Sub-function | Χρήστης | Δημιουργία Complement Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει μία Complement Κλάση |
| UC_32 | Sub-function | Χρήστης | Δημιουργία Απαριθμημένης Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει μία Απαριθμημένη Κλάση |
| UC_33 | Sub-function | Χρήστης | Δημιουργία Περιορισμού Ιδιότητας | Ο χρήστης επιθυμεί να δημιουργήσει έναν Περιορισμό για μία Ιδιότητα κάποιας Κλάσης |
| UC_34 | Sub-function | Χρήστης | Δημιουργία Αξιώματος Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει κάποιο αξίωμα για μία Κλάση |
| UC_35 | Primary Task | Χρήστης | Προσθήκη Datatype Property σε Κλάση | Ο χρήστης επιθυμεί να προσθέσει κάποιο Datatype Property σε μία Κλάση |
| UC_36 | Sub-function | Χρήστης | Δημιουργία Νέου Datatype Property | Ο χρήστης επιθυμεί να δημιουργήσει ένα νέο Datatype Property σε μία Κλάση |

| | | | | |
|---|---|---|---|---|
| UC_37 | Sub-function | Χρήστης | Προσθήκη Υπάρχοντος Datatype Property | Ο χρήστης επιθυμεί να προσθέσει κάποιο υπάρχον Datatype Property σε μία Κλάση |
| UC_38 | Sub-function | Χρήστης | Δημιουργία Αξιώματος Ιδιότητας | Ο χρήστης επιθυμεί να δημιουργήσει ένα αξίωμα για μία Ιδιότητα |
| UC_39 | Primary Task | Χρήστης | Δημιουργία Object Property | Ο χρήστης επιθυμεί να δημιουργήσει ένα Object Property |
| UC_40 | Sub-function | Χρήστης | Προσθήκη Object Property σε Κλάση | Ο χρήστης επιθυμεί να προσθέσει ένα Object Property σε μία Κλάση |
| UC_41 | Sub-function | Χρήστης | Ορισμός Τύπου Object Property | Ο χρήστης επιθυμεί να ορίσει τον τύπο ενός Object Property |
| UC_42 | Primary Task | Χρήστης | Δημιουργία Απαρίθμησης | Ο χρήστης επιθυμεί να δημιουργήσει μία Απαρίθμηση |
| UC_43 | Sub-function | Χρήστης | Δημιουργία Στοιχείου Απαρίθμησης | Ο χρήστης επιθυμεί να δημιουργήσει ένα στοιχείο για μία Απαρίθμηση |
| UC_44 | Primary Task | Χρήστης | Γραφικός Σχολιασμός Στοιχείου | Ο χρήστης επιθυμεί να δημιουργήσει ένα σχόλιο τύπου Comment σε ένα διάγραμμα |
| UC_45 | Sub-function | Χρήστης | Συσχετισμός Σχολίου-Στοιχείου | Ο χρήστης επιθυμεί να αντιστοιχίσει ένα σχόλιο τύπου Comment σε κάποιο στοιχείο που θέλει να σχολιάσει |
| UC_46 | Primary Task | Χρήστης | Δημιουργία Στιγμιότυπου | Ο χρήστης επιθυμεί να δημιουργήσει κάποιο Στιγμιότυπο |
| UC_47 | Sub-function | Χρήστης | Δημιουργία Στιγμιότυπου Κλάσης | Ο χρήστης επιθυμεί να δημιουργήσει ένα Στιγμιότυπο Κλάσης |
| UC_48 | Sub-function | Χρήστης | Δημιουργία Στιγμιότυπου Datatype Property | Ο χρήστης επιθυμεί να δημιουργήσει ένα Στιγμιότυπο Datatype Property |
| UC_49 | Sub-function | Χρήστης | Δημιουργία Στιγμιότυπου Object Property | Ο χρήστης επιθυμεί να δημιουργήσει ένα Στιγμιότυπο Object Property |
| UC_50 | Primary Task | Χρήστης | Διαγραφή Στοιχείου από Οντολογία | Ο χρήστης επιθυμεί να διαγράψει κάποιο στοιχείο της οντολογίας από την οντολογία |
| UC_51 | Sub-function | Χρήστης | Έλεγχος και Αποκατάσταση Συνέπειας | Το σύστημα ελέγχει και αποκαθιστά την συνέπεια της οντολογίας σε μία διαγραφή |
| UC_52 | Sub-function | Χρήστης | Διαγραφή Στοιχείου από Διάγραμμα | Διαγραφή ενός στοιχείου από κάποιο διάγραμμα. Η διαγραφή πραγματοποιείται από το χρήστη ή το σύστημα |
| UC_53 | Primary Task | Χρήστης | Τοποθέτηση Υπάρχοντος Στοιχείου σε Διάγραμμα | Ο χρήστης επιθυμεί να τοποθετήσει ένα υπάρχον στοιχείο της οντολογίας σε κάποιο διάγραμμα |
| UC_54 | Primary Task | Χρήστης | Επεξεργασία Χαρακτηριστικών Στοιχείου | Ο χρήστης επιθυμεί να επεξεργαστεί τα χαρακτηριστικά ενός στοιχείου της οντολογίας όπως αυτά ορίζονται από το ODM |

**Εικόνα 25: Συγκεντρωτικός Πίνακας των Περιπτώσεων Χρήσης**

Στη συνέχεια, παρουσιάζονται οι περιπτώσεις χρήσης που χρειάζεται να υποστηριχθούν κατά τη δημιουργία μιας οντολογίας. Κάθε περίπτωση χρήσης θα περιγράφεται με ένα πίνακα ο οποίος θα περιλαμβάνει όλες τις σχετικές με την περίπτωση χρήσης πληροφορίες. Το σύνολο των περιπτώσεων χρήσης που ακολουθεί δείχνει αναλυτικά αλλά και με κατανοητό τρόπο τη λειτουργικότητα που απαιτήθηκε και τελικά υποστηρίζει το εργαλείο (GRAMOFONE).

**Περίπτωση Χρήσης 1: Δημιουργία Οντολογίας.**

| | | |
|---|---|---|
| **USE CASE 1** | Δημιουργία Οντολογίας. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα ODM μοντέλο, δηλαδή μία οντολογία. | |
| **Scope & Level** | GRAMOFONE, Summary | |
| **Preconditions** | Ο χρήστης έχει «κατεβάσει» (download) το GRAMOFONE από την σελίδα του DBE και το έχει εγκαταστήσει στην πλατφόρμα Eclipse. | |
| **Success End Condition** | Ο χρήστης δημιουργεί την οντολογία. | |
| **Failed End Condition** | Η οντολογία δεν δημιουργείται, δεν πραγματοποιείται καμία μεταβολή. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων, Knowledge Base. | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία οντολογία χρησιμοποιώντας το ODM. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει με ποιο τρόπο επιθυμεί να δημιουργήσει την οντολογία. |
| | 2 | Ο χρήστης επεξεργάζεται τα μετά-δεδομένα της οντολογίας. (UC_3) |
| | 3 | Ο χρήστης επιχειρεί τη δημιουργία των όρων που επιθυμεί. |
| | 4 | Ο χρήστης κατά τη διάρκεια δημιουργίας των διαγραμμάτων μπορεί να τα διαχειριστεί. (UC_6) |
| | 5 | Ο χρήστης κατά την διάρκεια δημιουργίας της οντολογίας εξερευνεί τα περιεχόμενά της. (UC_15) |
| | 6 | Ο χρήστης είναι ανά πάσα στιγμή σε θέση να αποθηκεύσει την οντολογία. (UC_18) |
| | 7 | Ο χρήστης είναι ανά πάσα στιγμή σε θέση να διαγράψει κάποια οντολογία. (UC_21) |
| | 8 | Ο χρήστης ανά πάσα στιγμή μπορεί να διαγράψει κάποιο στοιχείο της οντολογίας. (UC_50) |
| | 9 | Ο χρήστης ανά πάσα στιγμή μπορεί να τοποθετήσει κάποιο υπάρχον στοιχείο της οντολογίας μέσα σε κάποιο διάγραμμα. (UC_53) |
| | 10 | Ο χρήστης ανά πάσα στιγμή μπορεί να επεξεργαστεί τα χαρακτηριστικά κάποιου στοιχείου της οντολογίας. (UC_54) |

| **Extensions** | **Step** | **Branching Action** |
|---|---|---|
| | 1α,2α, 3α,4α, 5α,6α, 7α,8α, 9α,10α | Δημιουργείται κάποιο σφάλμα στην λειτουργία. |
| | | Το GRAMOFONE ενημερώνει το χρήστη για το σφάλμα και την αιτία του όπου αυτό είναι δυνατό. |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει να δημιουργήσει μία νέα οντολογία (UC_2). |
| | 1β | Ο χρήστης επιλέγει να ανακτήσει μία οντολογία. (UC_24) |
| | 3α | Ο χρήστης δημιουργεί μία Κλάση. (UC_27) |
| | 3β | Ο χρήστης προσθέτει ένα Datatype Property σε μία Κλάση. (UC_35) |
| | 3γ | Ο χρήστης δημιουργεί ένα Object Property. (UC_39) |
| | 3δ | Ο χρήστης δημιουργεί μία Απαρίθμηση. (UC_42) |
| | 3ε | Ο χρήστης σχολιάζει γραφικά κάποιο στοιχείο. (UC_44) |
| | 3ζ | Ο χρήστης δημιουργεί ένα Στιγμιότυπο. (UC_46) |

**Περίπτωση Χρήσης 2: Δημιουργία Νέας Οντολογίας.**

| **USE CASE 2** | Δημιουργία Νέας Οντολογίας. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία καινούργια (δική του) οντολογία. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Η οντολογία δημιουργείται με επιτυχία, δημιουργούνται τα κατάλληλα αρχεία στο τοπικό σύστημα αρχείων και παρουσιάζεται από το GRAMOFONE ένα κενό (χωρίς στοιχεία) περιβάλλον δημιουργίας μίας νέας οντολογίας στην επιφάνεια εργασίας. | |
| **Failed End Condition** | Η οντολογία δεν δημιουργείται, το τοπικό σύστημα αρχείων παραμένει αμετάβλητο και η επιφάνεια εργασίας παραμένει επίσης αμετάβλητη. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό Σύστημα Αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE δημιουργίας νέας οντολογίας. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Εμφανίζεται ένα παράθυρο διαλόγου όπου ο χρήστης προτρέπεται να εισάγει το όνομα της νέας οντολογίας. |
| | 2 | Ο χρήστης εισάγει το όνομα της οντολογίας που επιθυμεί. |
| | 3 | Το σύστημα ελέγχει αν υπάρχει οντολογία στο χώρο εργασίας (workspace) του χρήστη με το ίδιο όνομα. |
| | 4 | Το σύστημα δημιουργεί στο χώρο εργασίας (workspace) ένα νέο έργο (project) με όνομα αυτό της οντολογίας. Η ενέργεια αυτή γίνεται μέσω της δημιουργίας ενός νέου φακέλου στο τοπικό σύστημα αρχείων. Εκεί θα αποθηκεύονται τα απαραίτητα αρχεία της οντολογίας. |
| | 5 | Το σύστημα ανοίγει την δημιουργηθείσα οντολογία στο |

| | | |
|---|---|---|
| | | περιβάλλον εργασίας (User Interface) και ενημερώνει με κατάλληλο μήνυμα το χρήστη για την επιτυχημένη αποπεράτωση της διαδικασίας (δημιουργία της οντολογίας). |
| **Extensions** | **Step** | **Branching Action** |
| | 3α | Υπάρχει οντολογία στο χώρο εργασίας (workspace) με το ίδιο όνομα. |
| | | 3ᵅ1. Το GRAMOFONE ακυρώνει τη λειτουργία δημιουργίας της νέας οντολογίας και εμφανίζει ένα μήνυμα που ενημερώνει το χρήστη για την ύπαρξη της οντολογίας στο χώρο εργασίας (workspace). |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ακυρώνει την λειτουργία. |
| | | 2ᵅ1. Η λειτουργία ακυρώνεται. Δεν πραγματοποιείται καμία αλλαγή. |

**Περίπτωση Χρήσης 3: Επεξεργασία Μεταδεδομένων Οντολογίας.**

| | | |
|---|---|---|
| **USE CASE 3** | Επεξεργασία Μεταδεδομένων Οντολογίας. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να τροποποιήσει τα χαρακτηριστικά της οντολογίας που επεξεργάζεται όπως αυτά ορίζονται από το ODM. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Τα χαρακτηριστικά της οντολογίας επεξεργάζονται με επιτυχία. | |
| **Failed End Condition** | Τα χαρακτηριστικά της οντολογίας δεν επεξεργάζονται. Το σύστημα παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να τροποποιήσει κάποιο από τα χαρακτηριστικά της οντολογίας που επεξεργάζεται. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά της οντολογίας. |
| | 2 | Ο χρήστης τροποποιεί τα χαρακτηριστικά της οντολογίας που επιθυμεί. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Τα χαρακτηριστικά της οντολογίας που μπορούν να τροποποιηθούν είναι τα παρακάτω. |
| | | 2ᵅ1. Το όνομα της οντολογίας. |
| | | 2ᵅ2. Ο συσχετισμός της υπό-επεξεργασία οντολογίας με κάποιες άλλες οντολογίες. (UC_4) |
| | | 2ᵅ3. Ο σχολιασμός της οντολογίας. (UC_5) |
| | | 2ᵅ4. Η εισαγωγή λιστών του τύπου "AllDifferent", όπως αυτές ορίζονται από το ODM. |

**Περίπτωση Χρήσης 4: Συσχετισμός Οντολογιών.**

| | | |
|---|---|---|
| **USE CASE 4** | Συσχετισμός Οντολογιών. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να συσχετίσει την οντολογία που επεξεργάζεται με άλλες οντολογίες. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Ο συσχετισμός της οντολογίας πραγματοποιείται με επιτυχία. | |
| **Failed End Condition** | Ο συσχετισμός της οντολογίας δεν πραγματοποιείται. Το σύστημα παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE, Βάση Γνώσης. | |
| **Trigger** | Ο χρήστης επιχειρεί να προσθέσει ένα νέο Ontology Property στην οντολογία που επεξεργάζεται. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Εμφανίζεται ένα παράθυρο διαλόγου (wizard) που προτρέπει το χρήστη να εισάγει την διεύθυνση (IP) που βρίσκεται η Βάση Γνώσης και να επιλέξει το όνομα του Ontology Property. |
| | 2 | Ο χρήστης συμπληρώνει τα πεδία. |
| | 3 | Το σύστημα προσπαθεί να συνδεθεί με την διεύθυνση που εισήγαγε ο χρήστης. |
| | 4 | Η σύνδεση επιτυγχάνεται και το σύστημα εμφανίζει σε μία λίστα τις διαθέσιμες οντολογίες και ο χρήστης παραπέμπεται να επιλέξει μία ή περισσότερες από αυτές. |
| | 5 | Ο χρήστης επιλέγει μία ή περισσότερες από τις διαθέσιμες οντολογίες. |
| | 6 | Το σύστημα δημιουργεί την Ontology Property, την παρουσιάζει στο περιβάλλον εργασίας (UI) και εμφανίζει μήνυμα που ενημερώνει τον χρήστη για την επιτυχία της λειτουργίας. |
| **Extensions** | **Step** | **Branching Action** |
| | 3α | Δημιουργείται σφάλμα κατά την σύνδεση. |
| | | $3^{\alpha}1$. Το σύστημα εμφανίζει ένα κατάλληλο μήνυμα και προτρέπει τον χρήστη είτε να επιστρέψει στην προηγούμενη σελίδα και να εισάγει ξανά την διεύθυνση ή να ακυρώσει την διαδικασία. |
| | | $3^{\alpha}1\alpha$. Ο χρήστης επιλέγει να επιστρέψει στην προηγούμενη σελίδα, οπότε μεταβαίνει στο βήμα 1. |
| | | $3^{\alpha}1\beta$. Ο χρήστης ακυρώνει την λειτουργία οπότε μεταβαίνει στο βήμα $2^{\alpha}$. |
| | 4α | Δεν υπάρχουν οντολογίες αποθηκευμένες στη Βάση Γνώσης και κατά συνέπεια ο χρήστης υποχρεώνεται να περάσει στο βήμα $2^{\alpha}$. |
| **Sub-Variations** | | **Branching Action** |
| | 1α, 2α, 5α | Ο χρήστης ακυρώνει την λειτουργία. |
| | | Η λειτουργία ακυρώνεται. Δεν πραγματοποιείται καμία αλλαγή. |
| | 5β | Ο χρήστης επιστρέφει στην προηγούμενη σελίδα ώστε να |

| | | |
|---|---|---|
| | | εισάγει μία νέα διεύθυνση. |
| | | 5β1. Μετάβαση στο βήμα 1. |

**Περίπτωση Χρήσης 5: Σχολιασμός Οντολογίας.**

| **USE CASE 5** | Σχολιασμός Οντολογίας. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να προσθέσει κάποιο σχόλιο στην οντολογία που επεξεργάζεται. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Δημιουργείται το σχόλιο της οντολογίας. | |
| **Failed End Condition** | Το σχόλιο δεν δημιουργείται. Το σύστημα παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να σχολιάσει την οντολογία που επεξεργάζεται. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα εμφανίζει ένα παράθυρο διαλόγου το οποίο προτρέπει το χρήστη να επιλέξει το είδος του σχολίου που επιθυμεί να εισάγει. |
| | 2 | Ο χρήστης επιλέγει το είδος του σχολίου. |
| | 3 | Το σύστημα προτρέπει το χρήστη να εισάγει το κείμενο του σχολίου. |
| | 4 | Ο χρήστης εισάγει το κείμενο του σχολίου. |
| | 5 | Το σύστημα δημιουργεί το σχόλιο και ενημερώνει το χρήστη με κατάλληλο μήνυμα για την προσθήκη του σχολίου. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α, 2α, 3α, 4α | Ο χρήστης ακυρώνει τη λειτουργία. |

**Περίπτωση Χρήσης 6: Διαχείριση Διαγραμμάτων Οντολογίας.**

| **USE CASE 6** | Διαχείριση Διαγραμμάτων Οντολογίας. |
|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να διαχειριστεί κάποιο από τα διαγράμματα της οντολογίας. |
| **Scope & Level** | GRAMOFONE, Primary Task |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο και πρέπει να υπάρχει μία οντολογία ενεργή στην επιφάνεια εργασίας. |
| **Success End Condition** | Η διαχείριση του διαγράμματος πραγματοποιείται με επιτυχία. |
| **Failed End Condition** | Η διαχείριση του διαγράμματος αποτυγχάνει. |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE. |

| | | |
|---|---|---|
| **Trigger** | Ο χρήστης επιχειρεί να διαχειριστεί κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει κάποια από τις λειτουργίες της διαχείρισης ενός διαγράμματος. |
| | 2 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει τη δημιουργία ενός διαγράμματος. (UC_7) |
| | 1β | Ο χρήστης επιλέγει τη μετονομασία ενός διαγράμματος. (UC_8) |
| | 1γ | Ο χρήστης επιλέγει τη διαγραφή ενός διαγράμματος. (UC_9) |
| | 1δ | Ο χρήστης επιλέγει την λειτουργία της εστίασης σε ένα διάγραμμα. (UC_10) |
| | 1ε | Ο χρήστης επιλέγει την λειτουργία της ευθυγράμμισης κάποιων στοιχείων ενός διαγράμματος. (UC_11) |
| | 1ζ | Ο χρήστης επιλέγει την ανάκτηση ενός διαγράμματος. (UC_12) |
| | 1η | Ο χρήστης επιλέγει την εκτύπωση ενός διαγράμματος. (UC_13) |
| | 1θ | Ο χρήστης επιλέγει την αυτόματη διάταξη των στοιχείων ενός διαγράμματος. (UC_14) |

**Περίπτωση Χρήσης 7: Δημιουργία Διαγράμματος.**

| | | |
|---|---|---|
| **USE CASE 7** | Δημιουργία Διαγράμματος. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα καινούργιο διάγραμμα στο πλαίσιο επεξεργασίας μίας οντολογίας. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Το διάγραμμα δημιουργείται με επιτυχία και ανοίγεται μέσα στο περιβάλλον εργασίας (UI). | |
| **Failed End Condition** | Το διάγραμμα δεν δημιουργείται. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE δημιουργίας διαγράμματος. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα ελέγχει αν υπάρχει κάποια οντολογία παρούσα στο περιβάλλον εργασίας (UI), δηλαδή να είναι ενεργή. |
| | 2 | Το σύστημα εμφανίζει ένα παράθυρο διαλόγου ζητώντας από το χρήστη να εισάγει το επιθυμητό όνομα του διαγράμματος. |

| | | |
|---|---|---|
| | 3 | Ο χρήστης εισάγει το επιθυμητό όνομα για το διάγραμμα. |
| | 4 | Το σύστημα ελέγχει αν υπάρχει διάγραμμα στην οντολογία με το ίδιο όνομα. |
| | 5 | Το σύστημα δημιουργεί στο χώρο εργασίας (workspace), μέσα στον ειδικό υποκατάλογο που αντιστοιχεί στην οντολογία, ένα αρχείο με όνομα το όνομα του διαγράμματος και επέκταση ".diagram". Στο αρχείο αυτό αποθηκεύονται τα στοιχεία του νέου διαγράμματος, δηλαδή αποτελεί ουσιαστικά το νέο διάγραμμα. |
| | 6 | Το σύστημα ανοίγει το νέο διάγραμμα στο περιβάλλον εργασίας για την περαιτέρω επεξεργασία του. |
| **Extensions** | **Step** | **Branching Action** |
| | 1α | Δεν υπάρχει ενεργή οντολογία στο περιβάλλον εργασίας. |
| | | 1α1. Το σύστημα εμφανίζει ένα μήνυμα που ενημερώνει τον χρήστη για την μη ύπαρξη ενεργής οντολογίας και τερματίζει την ενέργεια. |
| | 4α | Υπάρχει διάγραμμα στην οντολογία με το ίδιο όνομα. |
| | | 4α1. Το σύστημα εμφανίζει ένα μήνυμα που ενημερώνει τον χρήστη για την ύπαρξη διαγράμματος με την ίδια ονομασία και ακυρώνει (τερματίζει) την ενέργεια. |
| **Sub-Variations** | | **Branching Action** |
| | 3α | Ο χρήστης ακυρώνει την λειτουργία. |
| | | 3α1. Η λειτουργία ακυρώνεται. |

**Περίπτωση Χρήσης 8: Μετονομασία Διαγράμματος.**

| | | |
|---|---|---|
| **USE CASE 8** | Μετονομασία Διαγράμματος. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να μετονομάσει κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο και πρέπει να υπάρχει μία οντολογία ενεργή στην επιφάνεια εργασίας. | |
| **Success End Condition** | Το διάγραμμα μετονομάζεται με επιτυχία. | |
| **Failed End Condition** | Η μετονομασία αποτυγχάνει. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE μετονομασίας διαγράμματος. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα εμφανίζει ένα παράθυρο διαλόγου ζητώντας από το χρήστη να εισάγει το επιθυμητό όνομα του διαγράμματος. |
| | 2 | Ο χρήστης εισάγει το επιθυμητό όνομα για το διάγραμμα. |
| | 3 | Το σύστημα ελέγχει αν υπάρχει διάγραμμα στην οντολογία με το ίδιο όνομα. |
| | 4 | Το σύστημα μετονομάζει το διάγραμμα και ενημερώνει το χρήστη για την επιτυχημένη μετονομασία μέσω κατάλληλου μηνύματος.. |

| **Extensions** | **Step** | **Branching Action** |
|---|---|---|
| | 3α | Υπάρχει διάγραμμα στην οντολογία με το ίδιο όνομα. |
| | | 3$^{\alpha}$1. Το σύστημα εμφανίζει ένα μήνυμα που ενημερώνει το χρήστη για την ύπαρξη διαγράμματος με την ίδια ονομασία και ακυρώνει (τερματίζει) την ενέργεια. |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ακυρώνει τη λειτουργία. |
| | | 2$^{\alpha}$1. Η λειτουργία ακυρώνεται. |

**Περίπτωση Χρήσης 9: Διαγραφή Διαγράμματος.**

| **USE CASE 9** | Διαγραφή Διαγράμματος. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να διαγράψει κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο και πρέπει να υπάρχει μία οντολογία ενεργή στην επιφάνεια εργασίας. | |
| **Success End Condition** | Το διάγραμμα διαγράφεται με επιτυχία. | |
| **Failed End Condition** | Η διαγραφή αποτυγχάνει. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE διαγραφής διαγράμματος. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα εμφανίζει ένα παράθυρο διαλόγου ζητώντας από το χρήστη να επαληθεύσει τη διαγραφή του διαγράμματος. |
| | 2 | Ο χρήστης επαληθεύει τη διαγραφή του διαγράμματος. |
| | 3 | Το σύστημα διαγράφει το διάγραμμα, ανανεώνει την επιφάνεια εργασίας και ενημερώνει το χρήστη για την επιτυχημένη διαγραφή. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ακυρώνει τη λειτουργία. |
| | | 2$^{\alpha}$1. Η λειτουργία ακυρώνεται. |

**Περίπτωση Χρήσης 10: Εστίαση Σε Διάγραμμα.**

| **USE CASE 10** | Εστίαση Σε Διάγραμμα. |
|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να χρησιμοποιήσει τη λειτουργία της εστίασης σε ένα διάγραμμα. |
| **Scope & Level** | GRAMOFONE, Sub-function |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. Επιπλέον πρέπει να υπάρχει κάποιο διάγραμμα ανακτημένο στην επιφάνεια εργασίας. |
| **Success End Condition** | Η εστίαση επιτυγχάνεται. |

| | | |
|---|---|---|
| **Failed End Condition** | Η εστίαση αποτυγχάνει. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την λειτουργία της εστίασης ενός διαγράμματος που παρέχεται από το GRAMOFONE. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το είδος της εστίασης που επιθυμεί. |
| | 2 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει να εστιάσει προς τα μέσα. (Zoom In) |
| | 1β | Ο χρήστης επιλέγει να εστιάσει προς τα έξω. (Zoom Out) |
| | 1γ | Ο χρήστης επιλέγει να εστιάσει σε κάποιο συγκεκριμένο ποσοστό. Επιπλέον δίνονται οι δυνατότητες εστίασης ανά σελίδα, ανά ύψος και ανά πλάτος. |

**Περίπτωση Χρήσης 11: Ευθυγράμμιση Γραφικών Στοιχείων.**

| | | |
|---|---|---|
| **USE CASE 11** | Ευθυγράμμιση Γραφικών Στοιχείων. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να ευθυγραμμίσει κάποια από τα στοιχεία ενός διαγράμματος. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. Επιπλέον πρέπει να υπάρχει κάποιο διάγραμμα ανακτημένο στην επιφάνεια εργασίας. | |
| **Success End Condition** | Η ευθυγράμμιση επιτυγχάνεται. | |
| **Failed End Condition** | Η ευθυγράμμιση αποτυγχάνει. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί τη λειτουργία της ευθυγράμμισης δύο ή περισσοτέρων στοιχείων ενός διαγράμματος που παρέχεται από το GRAMOFONE. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει δύο ή περισσότερα στοιχεία πάνω στο διάγραμμα που επιθυμεί να ευθυγραμμίσει. |
| | 2 | Ο χρήστης επιλέγει το είδος της ευθυγράμμισης που επιθυμεί. |
| | 3 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. Τα στοιχεία ευθυγραμμίζονται γύρω από το στοιχείο που επιλέχθηκε τελευταίο σύμφωνα με την επιλογή του χρήστη. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |

| | | |
|---|---|---|
| | 2α | Οι δυνατές επιλογές ευθυγράμμισης που παρέχονται από το GRAMOFONE είναι αριστερά, δεξιά, στο κέντρο, στην κορυφή, στην βάση και στην μέση. |

**Περίπτωση Χρήσης 12: Ανάκτηση Διαγράμματος.**

| | | |
|---|---|---|
| **USE CASE 12** | Ανάκτηση Διαγράμματος. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να ανακτήσει στην επιφάνεια εργασίας κάποιο από τα διαγράμματα της οντολογίας που επεξεργάζεται. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Το διάγραμμα ανακτάται με επιτυχία και εμφανίζεται από το GRAMOFONE στην επιφάνεια εργασίας. | |
| **Failed End Condition** | Το διάγραμμα δεν ανακτάται και η επιφάνεια εργασίας παραμένει αμετάβλητη. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό Σύστημα Αρχείων. | |
| **Trigger** | Ο χρήστης επιλέγει το διάγραμμα που επιθυμεί να ανοίξει από την μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) και κάνει διπλό κλικ στο όνομα του διαγράμματος. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα ανοίγει το επιλεγμένο διάγραμμα στην επιφάνεια εργασίας (User Interface). |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 13: Εκτύπωση Διαγράμματος.**

| | | |
|---|---|---|
| **USE CASE 13** | Εκτύπωση Διαγράμματος. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να εκτυπώσει κάποιο διάγραμμα. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Η εκτύπωση πραγματοποιείται. | |
| **Failed End Condition** | Η εκτύπωση δεν πραγματοποιείται. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Εκτυπωτής τοπικού συστήματος. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την λειτουργία της εκτύπωσης ενός διαγράμματος που παρέχεται από το GRAMOFONE. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το διάγραμμα που επιθυμεί να εκτυπώσει και το ανακτά στην επιφάνεια εργασίας. (UC_12) |
| | 2 | Ο χρήστης επιλέγει την λειτουργία της εκτύπωσης. |
| | 3 | Το σύστημα εμφανίζει το προεπιλεγμένο (default) |

| | | |
|---|---|---|
| | | παράθυρο της εκτύπωσης εγγράφων. |
| | 4 | Ο χρήστης επαληθεύει την πρόθεσή του. |
| | 5 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας.. |
| **Extensions** | **Step** | **Branching Action** |
| | 5α | Η λειτουργία της εκτύπωσης αποτυγχάνει από εξωτερικούς λόγους και τερματίζει. |
| **Sub-Variations** | | **Branching Action** |
| | 4α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 14: Αυτόματη Διάταξη Γραφικών Στοιχείων.**

| **USE CASE 14** | Αυτόματη Διάταξη Γραφικών Στοιχείων. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να τοποθετηθούν αυτόματα από το GRAMOFONE τα στοιχεία ενός διαγράμματος. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Τα στοιχεία του διαγράμματος τοποθετούνται αυτόματα. | |
| **Failed End Condition** | Δεν πραγματοποιείται η αυτόματη τοποθέτηση των στοιχείων του διαγράμματος. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την λειτουργία της αυτόματης διάταξης των στοιχείων ενός διαγράμματος. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το διάγραμμα που επιθυμεί και το ανακτά στην επιφάνεια εργασίας. (UC_12) και επιλέγει την λειτουργία της αυτόματης διάταξης των στοιχείων του διαγράμματος. |
| | 2 | Το σύστημα τοποθετεί με τρόπο αυτόματο και βέλτιστο τα στοιχεία του διαγράμματος που είναι ανακτημένο στην επιφάνεια εργασίας. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 15: Εξερεύνηση Οντολογίας.**

| **USE CASE 15** | Εξερεύνηση Οντολογίας. |
|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να εξερευνήσει τα περιεχόμενα της οντολογίας που επεξεργάζεται. |
| **Scope & Level** | GRAMOFONE, Primary Task |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. |
| **Success End Condition** | Ο χρήστης εξερευνά με επιτυχία τα περιεχόμενα της οντολογίας. |

| | | |
|---|---|---|
| **Failed End Condition** | Η εξερεύνηση της οντολογίας αποτυγχάνει. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE. | |
| **Trigger** | Ο χρήστης επιχειρεί να εξερευνήσει τα περιεχόμενα της οντολογίας που επεξεργάζεται. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει κάποια από τις δυνατές λειτουργίες εξερεύνησης του περιεχομένου της οντολογίας. |
| | 2 | Το σύστημα αναλαμβάνει τη διεκπεραίωση της λειτουργίας που επιλέχθηκε. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει την αναζήτηση κάποιου όρου της οντολογίας. (UC_16) |
| | 1β | Ο χρήστης επιλέγει τον εντοπισμό ενός γραφικού στοιχείου σε ένα διάγραμμα. (UC_17) |

**Περίπτωση Χρήσης 16: Αναζήτηση Όρου Οντολογίας.**

| | | |
|---|---|---|
| **USE CASE 16** | Αναζήτηση Όρου Οντολογίας. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να αναζητήσει κάποιον από τους όρους που έχει ορίσει μέσα στην οντολογία που επεξεργάζεται. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Παρουσιάζονται οι όροι που ικανοποιούν τα κριτήρια αναζήτησης του χρήστη. | |
| **Failed End Condition** | Δεν υπάρχουν όροι οι οποίοι να ικανοποιούν τα κριτήρια αναζήτησης του χρήστη. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE. | |
| **Trigger** | Ο χρήστης επιλέγει και χρησιμοποιεί την φόρμα αναζήτησης όρων της οντολογίας. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης εισάγει κάποιο αλφαριθμητικό (String) το οποίο αντιστοιχεί σε τμήμα του ονόματος του όρου που αναζητά, και πατά το κουμπί αναζήτησης. |
| | 2 | Το σύστημα αναζητά τους όρους που περιέχουν στο όνομά τους το αλφαριθμητικό (String) που εισήγαγε ο χρήστης. |
| | 3 | Το σύστημα παρουσιάζει τους όρους που ικανοποιούν το κριτήριο αναζήτησης. |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Δεν υπάρχουν όροι που να αντιστοιχούν στο κριτήριο αναζήτησης. |
| | | $2^{\alpha}1$. Δεν εμφανίζονται αποτελέσματα και η λειτουργία τερματίζεται. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 17: Εντοπισμός Γραφικού Στοιχείου σε Διάγραμμα.**

| | | |
|---|---|---|
| **USE CASE 17** | Εντοπισμός Γραφικού Στοιχείου σε Διάγραμμα. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να εντοπίσει κάποιον όρο της οντολογίας που επεξεργάζεται μέσα σε ένα διάγραμμα. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Το στοιχείο εντοπίζεται με επιτυχία. | |
| **Failed End Condition** | Το στοιχείο δεν εντοπίζεται. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό Σύστημα Αρχείων. | |
| **Trigger** | Ο χρήστης επιλέγει το στοιχείο που επιθυμεί να εντοπίσει από την μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) και κάνει διπλό κλικ στο όνομα του στοιχείου. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα ανοίγει ένα από τα διαγράμματα που περιέχουν το επιλεγμένο στοιχείο στην επιφάνεια εργασίας και επιλέγει πάνω στο διάγραμμα το στοιχείο. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 18: Αποθήκευση Οντολογίας.**

| | | |
|---|---|---|
| **USE CASE 18** | Αποθήκευση Οντολογίας. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να αποθηκεύσει την οντολογία που επεξεργάζεται. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Επιτυγχάνεται η λειτουργία αποθήκευσης που επιλέγεται από τον χρήστη. | |
| **Failed End Condition** | Αποτυγχάνει η λειτουργία αποθήκευσης που επιλέγεται από τον χρήστη. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων, Βάση Γνώσης (Knowledge Base) | |
| **Trigger** | Ο χρήστης επιχειρεί να χρησιμοποιήσει κάποια από τις λειτουργίες αποθήκευσης που παρέχονται από το GRAMOFONE. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει κάποια από τις λειτουργίες αποθήκευσης. |
| | 2 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. |

| **Extensions** | **Step** | **Branching Action** |
|---|---|---|
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει να αποθηκεύσει την οντολογία στο τοπικό σύστημα αρχείων. (UC_19) |
| | 1β | Ο χρήστης επιλέγει να αποθηκεύσει την οντολογία στη Βάση Γνώσης. (UC_20). |

**Περίπτωση Χρήσης 19: Αποθήκευση Οντολογίας στο Τοπικό Σύστημα Αρχείων.**

| **USE CASE 19** | Αποθήκευση Οντολογίας στο Τοπικό Σύστημα Αρχείων. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να αποθηκεύσει την οντολογία που επεξεργάζεται στο τοπικό σύστημα αρχείων. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Η οντολογία σώζονται με επιτυχία. | |
| **Failed End Condition** | Τα οντολογία δεν σώζονται. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE αποθήκευσης της οντολογίας στο τοπικό σύστημα αρχείων. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα ελέγχει αν υπάρχουν διαγράμματα ενεργά στην επιφάνεια εργασίας. |
| | 2 | Το σύστημα ελέγχει αν κάποια από αυτά τα διαγράμματα χρειάζονται αποθήκευση. |
| | 3 | Το σύστημα ενημερώνει (συγχρονίζει) τα διαγράμματα που είναι αποθηκευμένα στο τοπικό σύστημα αρχείων αλλά όχι ανακτημένα στην επιφάνεια εργασίας. |
| | 4 | Το σύστημα ενημερώνει το αρχείο που κρατάει τις δομές που περιγράφουν τη σημασιολογία της οντολογίας τοπικά. |
| **Extensions** | **Step** | **Branching Action** |
| | 1α | Δεν υπάρχουν ενεργά διαγράμματα στην επιφάνεια εργασίας. |
| | | 1α1. Μετάβαση στο βήμα 3. |
| | 2α | Δεν υπάρχουν διαγράμματα που να χρειάζονται αποθήκευση. |
| | | 2α1. Μετάβαση στο βήμα 3. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 20: Αποθήκευση Οντολογίας στη Βάση Γνώσης.**

| **USE CASE 20** | Αποθήκευση Οντολογίας στη Βάση Γνώσης. |
|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να αποθηκεύσει την οντολογία που επεξεργάζεται στην επιφάνεια εργασίας (UI) στη Βάση Γνώσης. |
| **Scope & Level** | GRAMOFONE, Sub-function |

| | | |
|---|---|---|
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας.. | |
| **Success End Condition** | Η οντολογία αποθηκεύεται με επιτυχία στη Βάση Γνώσης. | |
| **Failed End Condition** | Η οντολογία δεν αποθηκεύεται στη Βάση Γνώσης. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Βάση Γνώσης. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE αποθήκευσης οντολογίας στη Βάση Γνώσης. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα εμφανίζει ένα παράθυρο διαλόγου που προτρέπει το χρήστη να εισάγει την διεύθυνση (IP) της Βάσης Γνώσης. |
| | 2 | Ο χρήστης εισάγει την διεύθυνση (IP) της Βάσης Γνώσης. |
| | 3 | Το σύστημα ελέγχει στην δοσμένη διεύθυνση και προσπαθεί να συνδεθεί σε αυτή. |
| | 4 | Το σύστημα ελέγχει αν υπάρχει οντολογία αποθηκευμένη στη Βάση Γνώσης με την ίδια ονομασία. |
| | 5 | Το σύστημα αποθηκεύει την οντολογία στη Βάση Γνώσης και εμφανίζει μήνυμα που ενημερώνει τον χρήστη για την επιτυχημένη αποθήκευση. |
| **Extensions** | **Step** | **Branching Action** |
| | 3α | Η διεύθυνση δεν είναι έγκυρη. |
| | | $3^{\alpha}1$. Η λειτουργία τερματίζεται. |
| | 4α | Υπάρχει οντολογία με την ίδια ονομασία αποθηκευμένη στη Βάση Γνώσης. |
| | | $4^{\alpha}1$. Ο χρήστης ερωτάται αν επιθυμεί να αντικατασταθεί η οντολογία της Βάσης Γνώσης. |
| | | $4^{\alpha}1\alpha$. Ο χρήστης επιλέγει την αντικατάσταση της οντολογίας της Βάσης Γνώσης από αυτή που επιθυμεί να αποθηκεύσει. |
| | | $4^{\alpha}1\alpha1$. Η οντολογία της επιφάνειας εργασίας αντικαθιστά την οντολογία της Βάσης Γνώσης και εμφανίζεται μήνυμα που ενημερώνει για την επιτυχημένη αντικατάσταση και η λειτουργία τερματίζει. |
| | | $4^{\alpha}1\beta$. Ο χρήστης ακυρώνει την λειτουργία επιλέγοντας να μην αντικατασταθεί η οντολογία της Βάσης Γνώσης. |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 21: Διαγραφή Οντολογίας.**

| | |
|---|---|
| **USE CASE 21** | Διαγραφή Οντολογίας. |
| **Goal in Context** | Ο χρήστης επιθυμεί να διαγράψει μία ή περισσότερες οντολογίες. |
| **Scope & Level** | GRAMOFONE, Primary Task |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. |
| **Success End Condition** | Επιτυγχάνεται η λειτουργία διαγραφής που επιλέγεται από τον χρήστη. |
| **Failed End Condition** | Αποτυγχάνει η λειτουργία διαγραφής που επιλέγεται από τον χρήστη. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. |

| | | |
|---|---|---|
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων, Βάση Γνώσης (Knowledge Base) | |
| **Trigger** | Ο χρήστης επιχειρεί να χρησιμοποιήσει κάποια από τις λειτουργίες διαγραφής που παρέχονται από το GRAMOFONE. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει κάποια από τις λειτουργίες διαγραφής. |
| | 2 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει να διαγράψει μία ή περισσότερες οντολογίες από το τοπικό σύστημα αρχείων. (UC_22). |
| | 1β | Ο χρήστης επιλέγει να διαγράψει μία ή περισσότερες οντολογίες από τη Βάση Γνώσης. (UC_23) |

**Περίπτωση Χρήσης 22: Διαγραφή Οντολογίας από το Τοπικό Σύστημα Αρχείων.**

| | | |
|---|---|---|
| **USE CASE 22** | Διαγραφή Οντολογίας από το Τοπικό Σύστημα Αρχείων. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να διαγράψει μία ή περισσότερες οντολογίες από το τοπικό σύστημα αρχείων. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Οι οντολογία (ή οντολογίες) διαγράφονται με επιτυχία. | |
| **Failed End Condition** | Δεν διαγράφεται καμία οντολογία. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE διαγραφής οντολογιών από τοπικό σύστημα αρχείων. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα εμφανίζει μία λίστα με όλες τις οντολογίες που αποθηκευμένες στο χώρο εργασίας (workspace) και προτρέπει τον χρήστη να επιλέξει ποιες από αυτές επιθυμεί να διαγράψει. |
| | 2 | Ο χρήστης επιλέγει μία ή περισσότερες από τις διαθέσιμες οντολογίες προς διαγραφή. |
| | 3 | Το σύστημα διαγράφει τις επιλεγμένες οντολογίες από το χώρο εργασίας (workspace) και εμφανίζει μήνυμα με το οποίο ενημερώνει τον χρήστη για την επιτυχημένη διαγραφή. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 23: Διαγραφή Οντολογίας από τη Βάση Γνώσης.**

| | | |
|---|---|---|
| **USE CASE 23** | Διαγραφή Οντολογίας από τη Βάση Γνώσης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να διαγράψει μία ή περισσότερες οντολογίες που είναι αποθηκευμένες στη Βάση Γνώσης. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Οι οντολογία (ή οντολογίες) διαγράφονται με επιτυχία. | |
| **Failed End Condition** | Δεν διαγράφεται καμία οντολογία. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Βάση Γνώσης (Knowledge Base). | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE διαγραφής οντολογιών από τη Βάση Γνώσης. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα εμφανίζει ένα παράθυρο διαλόγου το οποίο προτρέπει τον χρήστη να εισάγει την διεύθυνση (IP) της Βάσης Γνώσης. |
| | 2 | Ο χρήστης εισάγει την διεύθυνση (IP) της Βάσης Γνώσης. |
| | 3 | Το σύστημα ελέγχει την εισαγόμενη διεύθυνση και προσπαθεί να συνδεθεί σε αυτήν. |
| | 4 | Η σύνδεση επιτυγχάνεται και το σύστημα εμφανίζει σε μία λίστα όλες οι διαθέσιμες οντολογίες που είναι αποθηκευμένες στην Βάση Γνώσης. |
| | 5 | Ο χρήστης επιλέγει μία ή περισσότερες οντολογίες προς διαγραφή. |
| | 6 | Το σύστημα διαγράφει την επιλεγμένη οντολογία (ή οντολογίες) από την Βάση Γνώσης και εμφανίζει μήνυμα που ενημερώνει τον χρήστη για την πετυχημένη διαγραφή. |
| **Extensions** | **Step** | **Branching Action** |
| | 3α | Δημιουργείται σφάλμα κατά την σύνδεση. |
| | | $3^{\alpha}1$. Το σύστημα εμφανίζει μία σελίδα σφάλματος και προτρέπει τον χρήστη είτε να επιστρέψει στην προηγούμενη σελίδα και να εισάγει ξανά την διεύθυνση ή να ακυρώσει την διαδικασία. |
| | | $3^{\alpha}1$α. Ο χρήστης επιλέγει να επιστρέψει στην προηγούμενη σελίδα, οπότε μεταβαίνει στο βήμα 1. |
| | | $3^{\alpha}1$β. Ο χρήστης ακυρώνει την λειτουργία οπότε μεταβαίνει στο βήμα $2^{\alpha}$. |
| **Sub-Variations** | | **Branching Action** |
| | 2α, 5α | Ο χρήστης ακυρώνει την λειτουργία. |
| | 5β | Ο χρήστης επιστρέφει στην προηγούμενη σελίδα ώστε να εισάγει μία νέα διεύθυνση. |
| | | 5β1. Μετάβαση στο βήμα 1. |

**Περίπτωση Χρήσης 24: Ανάκτηση Οντολογίας.**

| | | |
|---|---|---|
| **USE CASE 24** | Ανάκτηση Οντολογίας. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να ανακτήσει κάποια υπάρχουσα οντολογία. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Επιτυγχάνεται η ανάκτηση της οντολογίας που επιλέγεται από το χρήστη. | |
| **Failed End Condition** | Η ανάκτηση της οντολογίας αποτυγχάνει. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων, Βάση Γνώσης (Knowledge Base) | |
| **Trigger** | Ο χρήστης επιχειρεί να χρησιμοποιήσει κάποια από τις λειτουργίες ανάκτησης οντολογιών που παρέχονται από το GRAMOFONE. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει κάποια από τις λειτουργίες ανάκτησης. |
| | 2 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει να ανακτήσει μία οντολογία από το τοπικό σύστημα αρχείων. (UC_25). |
| | 1β | Ο χρήστης επιλέγει να ανακτήσει μία οντολογία από τη Βάση Γνώσης. (UC_26) |

**Περίπτωση Χρήσης 25: Ανάκτηση Οντολογίας από το Τοπικό Σύστημα Αρχείων.**

| | | |
|---|---|---|
| **USE CASE 25** | Ανάκτηση Οντολογίας από το Τοπικό Σύστημα Αρχείων. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να ανακτήσει μία υπάρχουσα οντολογία που βρίσκεται αποθηκευμένη στο τοπικό σύστημα αρχείων, δηλαδή στο χώρο εργασίας (workspace) της πλατφόρμας του Eclipse. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Η οντολογία ανακτάται με επιτυχία και παρουσιάζεται στο περιβάλλον εργασίας (User Interface) του GRAMOFONE. | |
| **Failed End Condition** | Η οντολογία δεν ανακτάται. Το περιβάλλον εργασίας παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό Σύστημα Αρχείων. | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE ανάκτησης οντολογίας από το τοπικό σύστημα αρχείων. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Εμφανίζεται ένα παράθυρο διαλόγου που περιέχει τις διαθέσιμες οντολογίες και προτρέπει τον χρήστη να επιλέξει την οντολογία που επιθυμεί να ανακτηθεί. |
| | 2 | Ο χρήστης επιλέγει μία από τις διαθέσιμες οντολογίες. |

| | | |
|---|---|---|
| | 3 | Το σύστημα διαβάζει τις κατάλληλες δομές από το τοπικό σύστημα αρχείων, φορτώνει την οντολογία στην κύρια μνήμη, και την εμφανίζει στο περιβάλλον εργασίας. |
| **Extensions** | **Step** | **Branching Action** |
| | 1α | Δεν υπάρχουν οντολογίες αποθηκευμένες στο χώρο εργασίας (workspace) και κατά συνέπεια ο χρήστης υποχρεώνεται να περάσει στο βήμα $2^{α}$. |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ακυρώνει την λειτουργία. |
| | | $2^{α}1$. Η λειτουργία ακυρώνεται. Δεν πραγματοποιείται καμία αλλαγή. |

**Περίπτωση Χρήσης 26: Ανάκτηση Οντολογίας από τη Βάση Γνώσης.**

| | | |
|---|---|---|
| **USE CASE 26** | Ανάκτηση Οντολογίας από τη Βάση Γνώσης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να ανακτήσει μία υπάρχουσα οντολογία που βρίσκεται αποθηκευμένη στη Βάση Γνώσης. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Το GRAMOFONE πρέπει να είναι εγκατεστημένο. | |
| **Success End Condition** | Η οντολογία ανακτάται με επιτυχία και το περιβάλλον εργασίας διαμορφώνεται κατάλληλα. | |
| **Failed End Condition** | Η οντολογία δεν ανακτάται. Το περιβάλλον εργασίας (User Interface) παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>Τοπικό σύστημα αρχείων, Βάση Γνώσης (Knowledge Base). | |
| **Trigger** | Ο χρήστης χρησιμοποιεί την επιλογή του GRAMOFONE ανάκτησης οντολογίας από τη Βάση Γνώσης. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Εμφανίζεται ένα παράθυρο διαλόγου (wizard) που προτρέπει τον χρήστη να εισάγει την διεύθυνση (IP) που βρίσκεται η Βάση Γνώσης (Knowledge Base). |
| | 2 | Ο χρήστης εισάγει την διεύθυνση που βρίσκεται η Βάση Γνώσης. |
| | 3 | Το σύστημα προσπαθεί να συνδεθεί με την διεύθυνση που εισήγαγε ο χρήστης. |
| | 4 | Η σύνδεση επιτυγχάνεται και εμφανίζονται σε μία λίστα οι διαθέσιμες οντολογίες και ο χρήστης παραπέμπεται να επιλέξει μία από αυτές. |
| | 5 | Ο χρήστης επιλέγει μία από τις διαθέσιμες οντολογίες. |
| | 6 | Το σύστημα ελέγχει αν υπάρχει αποθηκευμένη στο χώρο εργασίας (workspace) οντολογία με το ίδιο όνομα με την οντολογία που επιλέχθηκε. |
| | 7 | Το σύστημα ανακτά την επιλεγμένη οντολογία, την αποθηκεύει στο χώρο εργασίας (workspace), την παρουσιάζει στο περιβάλλον εργασίας (UI) και εμφανίζει μήνυμα που ενημερώνει τον χρήστη για την πετυχημένη ανάκτηση της οντολογίας. |
| **Extensions** | **Step** | **Branching Action** |

| | | |
|---|---|---|
| | 3α | Δημιουργείται σφάλμα κατά την σύνδεση. |
| | | 3$^{α}$1. Το σύστημα εμφανίζει ένα κατάλληλο μήνυμα και προτρέπει τον χρήστη είτε να επιστρέψει στην προηγούμενη σελίδα και να εισάγει ξανά την διεύθυνση ή να ακυρώσει την διαδικασία. |
| | | 3$^{α}$1α. Ο χρήστης επιλέγει να επιστρέψει στην προηγούμενη σελίδα, οπότε μεταβαίνει στο βήμα 1. |
| | | 3$^{α}$1β. Ο χρήστης ακυρώνει την λειτουργία οπότε μεταβαίνει στο βήμα 2$^{α}$. |
| | 4α | Δεν υπάρχουν οντολογίες αποθηκευμένες στη Βάση Γνώσης και κατά συνέπεια ο χρήστης υποχρεώνεται να περάσει στο βήμα 2$^{α}$. |
| | 6α | Υπάρχει οντολογία με το ίδιο όνομα στο τοπικό σύστημα αρχείων. |
| | | 6$^{α}$1. Το σύστημα ενημερώνει τον χρήστη για την υπάρχουσα οντολογία και τον προτρέπει να επιλέξει εάν επιθυμεί να προχωρήσει στην αντικατάσταση της οντολογίας που υπάρχει στο τοπικό σύστημα αρχείων με αυτήν που υπάρχει στη Βάση Γνώσης. |
| | | 6$^{α}$1α. Ο χρήστης επιλέγει την αντικατάσταση της οντολογίας που βρίσκεται στο τοπικό σύστημα αρχείων. |
| | | 6$^{α}$1α1. Το σύστημα διαγράφει την οντολογία που βρίσκεται στο τοπικό σύστημα αρχείων και προχωράει στο βήμα 7. |
| | | 6$^{α}$1β. Ο χρήστης επιλέγει να μην αντικαταστήσει την οντολογία που βρίσκεται στο τοπικό σύστημα αρχείων. |
| | | 6$^{α}$1β1. Η λειτουργία ακυρώνεται. Δεν πραγματοποιείται καμία αλλαγή. |
| **Sub-Variations** | | **Branching Action** |
| | 1α, 2α, 5α | Ο χρήστης ακυρώνει την λειτουργία. |
| | | Η λειτουργία ακυρώνεται. Δεν πραγματοποιείται καμία αλλαγή. |
| | 5β | Ο χρήστης επιστρέφει στην προηγούμενη σελίδα ώστε να εισάγει μία νέα διεύθυνση. |
| | | 5β1. Μετάβαση στο βήμα 1. |

**Περίπτωση Χρήσης 27: Δημιουργία Κλάσης.**

| | | |
|---|---|---|
| **USE CASE 27** | Δημιουργία Κλάσης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να ορίσει μία Κλάση. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Ορίζεται μία νέα Κλάση. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |

| | | |
|---|---|---|
| | 1 | Ο χρήστης επιλέγει το στοιχείο «Κλάση» από την παλέτα του διαγράμματος και το τοποθετεί μέσα στο διάγραμμα. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| | 6 | Ο χρήστης ορίζει περαιτέρω την Κλάση. |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | $2^{\alpha}1$. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | 6α | Ο χρήστης δημιουργεί μία Named Κλάση. (UC_28) |
| | 6β | Ο χρήστης δημιουργεί μία Union Κλάση. (UC_29) |
| | 6γ | Ο χρήστης δημιουργεί μία Intersection Κλάση. (UC_30) |
| | 6δ | Ο χρήστης δημιουργεί μία Complement Κλάση. (UC_31) |
| | 6ε | Ο χρήστης δημιουργεί μία Απαριθμημένη Κλάση. (UC_32) |
| | 6ζ | Ο χρήστης δημιουργεί ένα Περιορισμό σε μία Ιδιότητα της Κλάσης. (UC_33) |
| | 6η | Ο χρήστης δημιουργεί ένα αξίωμα για την Κλάση. (UC_34) |

**Περίπτωση Χρήσης 28: Δημιουργία Named Κλάσης.**

| | | |
|---|---|---|
| **USE CASE 28** | Δημιουργία Named Κλάσης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία Named Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) | |
| **Success End Condition** | Η named Κλάση δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Η named Κλάση δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Named Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης τροποποιεί το όνομα της Κλάσης της οντολογίας που επιθυμεί. |
| | 3 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 4 | Δημιουργείται από το σύστημα η Named Κλάση. |
| **Extensions** | **Step** | **Branching Action** |

| | | |
|---|---|---|
| | 3α | Υπάρχει Κλάση με το ίδιο όνομα. |
| | | 3$^{\alpha}$1. Η λειτουργία ακυρώνεται και ο χρήστης ενημερώνεται για την ύπαρξη και δεύτερης Κλάσης με το ίδιο όνομα στην οντολογία. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 29: Δημιουργία Union Κλάσης.**

| | | |
|---|---|---|
| **USE CASE 29** | Δημιουργία Union Κλάσης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία Union Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) | |
| **Success End Condition** | Η Union Κλάση δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Η Union Κλάση δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Union Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη μίας Union Κλάσης. |
| | 3 | Το σύστημα εμφανίζει μία λίστα με τις διαθέσιμες Κλάσεις της οντολογίας. |
| | 4 | Ο χρήστης επιλέγει την Κλάση που επιθυμεί. |
| | 5 | Το σύστημα προσθέτει την επιλεγμένη Κλάση στις Union Κλάσεις της Κλάσης που επεξεργάζεται ο χρήστης και ενημερώνει με κατάλληλο μήνυμα το χρήστη για την πετυχημένη προσθήκη. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 4α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 30: Δημιουργία Intersection Κλάσης.**

| | |
|---|---|
| **USE CASE 30** | Δημιουργία Intersection Κλάσης. |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία Intersection Κλάση. |
| **Scope & Level** | GRAMOFONE, Sub-function |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) |
| **Success End Condition** | Η Intersection Κλάση δημιουργείται με επιτυχία. |
| **Failed End Condition** | Η Intersection Κλάση δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. |
| **Primary,** | Χρήστης, |

| **Secondary Actors** | GRAMOFONE | |
|---|---|---|
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Intersection Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη μίας Intersection Κλάσης. |
| | 3 | Το σύστημα εμφανίζει μία λίστα με τις διαθέσιμες Κλάσεις της οντολογίας. |
| | 4 | Ο χρήστης επιλέγει την Κλάση που επιθυμεί. |
| | 5 | Το σύστημα προσθέτει την επιλεγμένη Κλάση στις Intersection Κλάσεις της Κλάσης που επεξεργάζεται ο χρήστης και ενημερώνει με κατάλληλο μήνυμα το χρήστη για την πετυχημένη προσθήκη. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 4α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 31: Δημιουργία Complement Κλάσης.**

| **USE CASE 31** | Δημιουργία Complement Κλάσης. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία Complement Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) | |
| **Success End Condition** | Η Complement Κλάση δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Η Complement Κλάση δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Complement Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη μίας Complement Κλάσης. |
| | 3 | Το σύστημα εμφανίζει μία λίστα με τις διαθέσιμες Κλάσεις της οντολογίας. |
| | 4 | Ο χρήστης επιλέγει την Κλάση που επιθυμεί. |
| | 5 | Το σύστημα προσθέτει την επιλεγμένη Κλάση σαν Complement της Κλάσης που επεξεργάζεται ο χρήστης και ενημερώνει με κατάλληλο μήνυμα το χρήστη για την πετυχημένη προσθήκη. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 4α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 32: Δημιουργία Απαριθμημένης Κλάσης.**

| **USE CASE 32** | Δημιουργία Απαριθμημένης Κλάσης. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία Απαριθμημένη Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) | |
| **Success End Condition** | Η Απαριθμημένη Κλάση δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Η Απαριθμημένη Κλάση δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Απαριθμημένη Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη μίας Απαριθμημένης Κλάσης. |
| | 3 | Το σύστημα εμφανίζει μία λίστα με τα διαθέσιμα Στιγμιότυπα Κλάσεων της οντολογίας. |
| | 4 | Ο χρήστης επιλέγει τα Στιγμιότυπα Κλάσεων που επιθυμεί. |
| | 5 | Το σύστημα προσθέτει τα επιλεγμένα Στιγμιότυπα Κλάσεων στα Στιγμιότυπα της Κλάσης που επεξεργάζεται ο χρήστης και ενημερώνει με κατάλληλο μήνυμα το χρήστη για την πετυχημένη προσθήκη. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 4α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 33: Δημιουργία Περιορισμού Ιδιότητας.**

| **USE CASE 33** | Δημιουργία Περιορισμού Ιδιότητας. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει έναν Περιορισμό για μία Ιδιότητα κάποιας Κλάσης. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) | |
| **Success End Condition** | Ο Περιορισμός δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Ο Περιορισμός δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει έναν Περιορισμό για μία Ιδιότητα μιας Κλάσης. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά |

| | | |
|---|---|---|
| | | μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη ενός Περιορισμού σε μία Ιδιότητα της Κλάσης που έχει επιλέξει. |
| | 3 | Το σύστημα εμφανίζει μία λίστα με τις διαθέσιμες Ιδιότητες της Κλάσης που έχει επιλέξει ο χρήστης. |
| | 4 | Ο χρήστης επιλέγει την Ιδιότητα που επιθυμεί να θέσει τον Περιορισμό. |
| | 5 | Ο χρήστης επιλέγει τους περιορισμούς που επιθυμεί να θέσει στην συγκεκριμένη Ιδιότητα. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 5α | Τα χαρακτηριστικά του Περιορισμού που ορίζονται στο ODM είναι τα ακόλουθα: "onProperty", "minCardinality", "maxCardinality", "cardinality", "allValuesFrom", "someValuesFrom" και "hasValue". |

**Περίπτωση Χρήσης 34: Δημιουργία Αξιώματος Κλάσης.**

| | | |
|---|---|---|
| **USE CASE 34** | Δημιουργία Αξιώματος Κλάσης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει κάποιο αξίωμα για μία Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία και να έχει δημιουργηθεί μία Κλάση. (UC_27) | |
| **Success End Condition** | Το αξίωμα δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Το αξίωμα δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει κάποιο αξίωμα για μία Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Κλάσης της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη ενός αξιώματος για την Κλάση που έχει επιλέξει. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης ορίζει την Κλάση ως Deprecated. |
| | 2β | Ο χρήστης προσθέτει μία ή περισσότερες Κλάσεις ως ισοδύναμες της Κλάσης που έχει επιλέξει. |
| | 2γ | Ο χρήστης προσθέτει μία ή περισσότερες Κλάσεις ως disjointed Κλάσεις της Κλάσης που έχει επιλέξει. |
| | 2δ | Ο χρήστης ορίζει μία Κλάση ως υπό-κλάση της Κλάσης που έχει επιλέξει. |
| | 2ε | Ο χρήστης ορίζει μία Κλάση ως υπέρ-κλάση της Κλάσης που έχει επιλέξει. |

**Περίπτωση Χρήσης 35: Προσθήκη Datatype Property σε Κλάση.**

| | | |
|---|---|---|
| **USE CASE 35** | Προσθήκη Datatype Property σε Κλάση. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να προσθέσει κάποιο Datatype Property σε μία Κλάση. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Προστίθεται ένα Datatype Property στην Κλάση που επιλέγει ο χρήστης. | |
| **Failed End Condition** | Η οντολογία παραμένει αμετάβλητη. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να προσθέσει κάποιο Datatype Property σε μία Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει μία Κλάση μέσα σε ένα διάγραμμα και επιλέγει τον τρόπο με τον οποίο επιθυμεί να προσθέσει ένα Datatype Property στην Κλάση. |
| | 2 | Ο χρήστης δημιουργεί κάποιο αξίωμα για την Ιδιότητα. (UC_38) |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει τη δημιουργία νέου Datatype Property. (UC_36) |
| | 1β | Ο χρήστης επιλέγει την προσθήκη ενός υπάρχοντος Datatype Property. (UC_37) |

**Περίπτωση Χρήσης 36: Δημιουργία Νέου Datatype Property.**

| | | |
|---|---|---|
| **USE CASE 36** | Δημιουργία Νέου Datatype Property. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα νέο Datatype Property σε μία Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Δημιουργείται ένα νέο Datatype Property. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα νέο Datatype Property σε μία Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |

| | | |
|---|---|---|
| | 1 | Ο χρήστης επιλέγει μία Κλάση μέσα σε ένα διάγραμμα και επιλέγει την προσθήκη ενός νέου Datatype Property για την Κλάση. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος). |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2α1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Υπάρχουν τρεις δυνατές επιλογές δημιουργίας ενός Datatype Property σε μία Κλάση. Μέσω της γραμμής εργαλείων (tool bar) του GRAMOFONE, με δεξί κλικ πάνω στην Κλάση γραφικά, και από την μονάδα προβολής των χαρακτηριστικών της Κλάσης (Property Sheet). |

**Περίπτωση Χρήσης 37: Προσθήκη Υπάρχοντος Datatype Property.**

| | | |
|---|---|---|
| **USE CASE 37** | Προσθήκη Υπάρχοντος Datatype Property. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να προσθέσει κάποιο υπάρχον Datatype Property σε μία Κλάση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Προστίθεται ένα υπάρχον Datatype Property στην Κλάση. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να προσθέσει κάποιο υπάρχον Datatype Property σε μία Κλάση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει μία Κλάση μέσα σε ένα διάγραμμα και επιλέγει την προσθήκη ενός υπάρχοντος Datatype Property στην Κλάση. |
| | 2 | Το σύστημα εμφανίζει μία λίστα με τα διαθέσιμα Datatype Properties της οντολογίας. |
| | 3 | Ο χρήστης επιλέγει το Datatype Property που επιθυμεί. |
| | 4 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 5 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 6 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 7 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης |

| | | |
|---|---|---|
| | | του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος). |
| **Extensions** | **Step** | **Branching Action** |
| | 4α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 4$^{\alpha}$1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Εναλλακτικά η λειτουργία μπορεί να πραγματοποιηθεί μέσω Drag And Drop (DND) μεταξύ δύο Κλάσεων. (η source Κλάση πρέπει να περιέχει Datatype Properties) |
| | 3α | Ο χρήστης ακυρώνει τη λειτουργία. |

**Περίπτωση Χρήσης 38: Δημιουργία Αξιώματος Ιδιότητας.**

| | | |
|---|---|---|
| **USE CASE 38** | Δημιουργία Αξιώματος Ιδιότητας. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει κάποιο αξίωμα για μία Ιδιότητα. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Το αξίωμα δημιουργείται με επιτυχία. | |
| **Failed End Condition** | Το αξίωμα δεν δημιουργείται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει κάποιο αξίωμα για μία Ιδιότητα. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά μίας Ιδιότητας της οντολογίας. |
| | 2 | Ο χρήστης επιλέγει την προσθήκη ενός αξιώματος για την Ιδιότητα που έχει επιλέξει. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει ένα Datatype Property. |
| | 1β | Ο χρήστης επιλέγει ένα Object Property. |
| | 2α | Η Ιδιότητα είναι Datatype Property. |
| | | 2$^{\alpha}$1. Η Ιδιότητα ορίζεται ως Deprecated. |
| | | 2$^{\alpha}$2. Η Ιδιότητα ορίζεται ως Functional. |
| | 2β | Η Ιδιότητα είναι Object Property. |
| | | 2β1. Η Ιδιότητα ορίζεται ως Transitive. |
| | | 2β2. Η Ιδιότητα ορίζεται ως Symmetric. |
| | | 2β3. Η Ιδιότητα ορίζεται ως Functional. |
| | | 2β4. Η Ιδιότητα ορίζεται ως InverseFunctional. |
| | | 2β5. Η Ιδιότητα ορίζεται ως Deprecated. |

**Περίπτωση Χρήσης 39: Δημιουργία Object Property.**

| | | |
|---|---|---|
| **USE CASE 39** | Δημιουργία Object Property. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα Object Property. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Δημιουργείται ένα νέο Object Property. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα Object Property. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο «Ιδιότητα τύπου Object Property» από την παλέτα του διαγράμματος και το τοποθετεί μέσα στο διάγραμμα. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| | 6 | Ο χρήστης δημιουργεί ένα αξίωμα για την Ιδιότητα. (UC_38) |
| | 7 | Ο χρήστης προσθέτει το Object Property σε μία Κλάση. (UC_40) |
| | 8 | Ο χρήστης ορίζει τον τύπο του Object Property. (UC_41) |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2ᵅ1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 40: Προσθήκη Object Property σε Κλάση.**

| | |
|---|---|
| **USE CASE 40** | Προσθήκη Object Property σε Κλάση. |
| **Goal in Context** | Ο χρήστης επιθυμεί να προσθέσει μία Domain Κλάση σε ένα Object Property. |
| **Scope & Level** | GRAMOFONE, Sub-function |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) |
| **Success End Condition** | Προστίθεται μία Domain Κλάση για το επιλεγμένο Object Property. |

| | | |
|---|---|---|
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να προσθέσει μία Domain Κλάση σε ένα Object Property. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο της σύνδεσης «Ορισμός Domain Κλάσης» από την παλέτα του διαγράμματος και επιχειρεί να συνδέσει μέσα στο διάγραμμα μία Κλάση και μία Ιδιότητα τύπου Object Property (δηλαδή την Domain Κλάση με την Ιδιότητα). |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Η σύνδεση εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη της σύνδεσης. |
| | 5 | Το GRAMOFONE ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2α1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 41: Ορισμός Τύπου Object Property.**

| | | |
|---|---|---|
| **USE CASE 41** | Ορισμός Τύπου Object Property. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να προσθέσει μία Range Κλάση σε ένα Object Property. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Προστίθεται μία Range Κλάση για το επιλεγμένο Object Property. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να προσθέσει μία Range Κλάση σε ένα Object Property. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο της σύνδεσης «Ορισμός Range |

| | | |
|---|---|---|
| | | Κλάσης» από την παλέτα του διαγράμματος και επιχειρεί να συνδέσει μέσα στο διάγραμμα μία Ιδιότητα τύπου Object Property και μία Κλάση (δηλαδή την Ιδιότητα με την Range Κλάση). |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Η σύνδεση εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη της σύνδεσης. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2α1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 42: Δημιουργία Απαρίθμησης.**

| | | |
|---|---|---|
| **USE CASE 42** | Δημιουργία Απαρίθμησης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει μία Απαρίθμηση. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Δημιουργείται μία νέα Απαρίθμηση. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει μία Απαρίθμηση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο «Απαρίθμηση» (Enumeration) από την παλέτα του διαγράμματος και το τοποθετεί μέσα στο διάγραμμα. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| | 6 | Ο χρήστης επιλέγει να δημιουργήσει ένα στοιχείο της Απαρίθμησης. (UC_43) |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του |

| | | |
|---|---|---|
| | | χρήστη. |
| | | 2ᵅ1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 43: Δημιουργία Στοιχείου Απαρίθμησης.**

| | | |
|---|---|---|
| **USE CASE 43** | Δημιουργία Στοιχείου Απαρίθμησης. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα στοιχείο για μία Απαρίθμηση. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Δημιουργείται ένα νέο στοιχείο για την Απαρίθμηση. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα στοιχείο για μία Απαρίθμηση. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει μία Απαρίθμηση (Enumeration) μέσα σε ένα διάγραμμα και επιλέγει την προσθήκη ενός νέου Plain Literal για την Απαρίθμηση αυτή. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2ᵅ1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Υπάρχουν τρεις δυνατές επιλογές ορισμού ενός Plain Literal για μία Απαρίθμηση. Μέσω της γραμμής εργαλείων (tool bar) του GRAMOFONE, με δεξί κλικ πάνω Απαρίθμηση (Enumeration) γραφικά, και από την μονάδα προβολής των χαρακτηριστικών της Απαρίθμησης (Property Sheet). |

**Περίπτωση Χρήσης 44: Γραφικός Σχολιασμός Στοιχείου.**

| | | |
|---|---|---|
| **USE CASE 44** | Γραφικός Σχολιασμός Στοιχείου. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα Σχόλιο τύπου Comment σε ένα διάγραμμα. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Δημιουργείται ένα νέο Σχόλιο τύπου Comment. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα Σχόλιο τύπου Comment. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο «Σχόλιο» (Comment) από την παλέτα του διαγράμματος και το τοποθετεί μέσα στο διάγραμμα. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| | 6 | Ο χρήστης συσχετίζει το Σχόλιο με το στοιχείο που επιθυμεί να σχολιάσει. (UC_45) |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | $2^{\alpha}1$. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 45: Συσχετισμός Σχολίου - Στοιχείου.**

| | |
|---|---|
| **USE CASE 45** | Συσχετισμός Σχολίου – Στοιχείου. |
| **Goal in Context** | Ο χρήστης επιθυμεί να αντιστοιχίσει ένα σχόλιο τύπου Comment σε κάποιο στοιχείο που θέλει να σχολιάσει. |
| **Scope & Level** | GRAMOFONE, Sub-function |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) |
| **Success End Condition** | Αντιστοιχίζεται το Σχόλιο στο στοιχείο που επιθυμεί ο χρήστης να σχολιάσει. |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. |
| **Primary,** | Χρήστης, |

| | | |
|---|---|---|
| **Secondary Actors** | GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να αντιστοιχήσει ένα Σχόλιο στο στοιχείο που επιθυμεί να σχολιάσει. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο της σύνδεσης «Αντιστοίχιση Σχολίου» από την παλέτα του διαγράμματος και επιχειρεί να συνδέσει μέσα στο διάγραμμα ένα Σχόλιο τύπου Comment και το στοιχείο που επιθυμεί να σχολιάσει. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Η σύνδεση εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη της σύνδεσης. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2α1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 46: Δημιουργία Στιγμιότυπου.**

| | | |
|---|---|---|
| **USE CASE 46** | Δημιουργία Στιγμιότυπου. | |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει κάποιο Στιγμιότυπο. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει κάποια ενεργή οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Επιτυγχάνεται η δημιουργία του Στιγμιότυπου που επιλέγεται από τον χρήστη. | |
| **Failed End Condition** | Αποτυγχάνει η δημιουργία του Στιγμιότυπου που επιλέγεται από τον χρήστη. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει κάποιο Στιγμιότυπο. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει ποιο είδος Στιγμιότυπου επιθυμεί να δημιουργήσει. |
| | 2 | Το σύστημα αναλαμβάνει την διεκπεραίωση της επιλεγμένης λειτουργίας. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει να δημιουργήσει ένα Στιγμιότυπο |

| | | |
|---|---|---|
| | | Κλάσης. (UC_47). |
| | 1β | Ο χρήστης επιλέγει να δημιουργήσει ένα Στιγμιότυπο Datatype Property. (UC_48) |
| | 1γ | Ο χρήστης επιλέγει να δημιουργήσει ένα Στιγμιότυπο Object Property. (UC_49) |

**Περίπτωση Χρήσης 47: Δημιουργία Στιγμιότυπου Κλάσης.**

| **USE CASE 47** | Δημιουργία Στιγμιότυπου Κλάσης. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα Στιγμιότυπο Κλάσης. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) | |
| **Success End Condition** | Δημιουργείται ένα νέο Στιγμιότυπο Κλάσης (Class Thing). | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα Στιγμιότυπο Κλάσης. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει το στοιχείο «Στιγμιότυπο Κλάσης» (Class Thing) από την παλέτα του διαγράμματος και το τοποθετεί μέσα στο διάγραμμα. |
| | 2 | Το σύστημα εμφανίζει ένα παράθυρο που προτρέπει τον χρήστη να ορίσει το όνομα του Στιγμιότυπου καθώς και την Κλάση της οποίας αποτελεί στιγμιότυπο. |
| | 3 | Ο χρήστης συμπληρώνει τα απαραίτητα στοιχεία. |
| | 4 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 5 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 6 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 3α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 48: Δημιουργία Στιγμιότυπου Datatype Property.**

| **USE CASE 48** | Δημιουργία Στιγμιότυπου Datatype Property. |
|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα Στιγμιότυπο Datatype Property. |
| **Scope & Level** | GRAMOFONE, Sub-function |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) |

| | | |
|---|---|---|
| **Success End Condition** | Δημιουργείται ένα νέο Στιγμιότυπο για μία Ιδιότητα τύπου Datatype Property. | |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα Στιγμιότυπο Datatype Property. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει ένα Στιγμιότυπο μίας Κλάσης (Class Thing) μέσα σε ένα διάγραμμα και επιλέγει την προσθήκη ενός νέου Στιγμιότυπου μίας Ιδιότητας Datatype Property. |
| | 2 | Το σύστημα ελέγχει την επιλογή του χρήστη. |
| | 3 | Το στοιχείο εμφανίζεται μέσα στο διάγραμμα. |
| | 4 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 5 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Υπάρχει περιορισμός που απαγορεύει την επιλογή του χρήστη. |
| | | 2$^{\alpha}$1. Η ενέργεια αποτρέπεται μέσω φιλικής προς τον χρήστη υπόδειξης. |
| **Sub-Variations** | | **Branching Action** |
| | 1$^{\alpha}$ | Υπάρχουν τρεις δυνατές επιλογές ορισμού ενός Στιγμιότυπου μίας Ιδιότητας Datatype Property. Μέσω της γραμμής εργαλείων (tool bar) του GRAMOFONE, με δεξί κλικ πάνω στο Στιγμιότυπο της Κλάσης (Class Thing) γραφικά, και από την μονάδα προβολής των χαρακτηριστικών του Στιγμιότυπου της Κλάσης (Property Sheet). |

**Περίπτωση Χρήσης 49: Δημιουργία Στιγμιότυπου Object Property.**

| | |
|---|---|
| **USE CASE 49** | Δημιουργία Στιγμιότυπου Object Property. |
| **Goal in Context** | Ο χρήστης επιθυμεί να δημιουργήσει ένα Στιγμιότυπο Object Property. |
| **Scope & Level** | GRAMOFONE, Sub-function |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) |
| **Success End Condition** | Δημιουργείται ένα νέο Στιγμιότυπο Object Property. |
| **Failed End Condition** | Το διάγραμμα παραμένει αμετάβλητο. |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE |
| **Trigger** | Ο χρήστης επιχειρεί να δημιουργήσει ένα Στιγμιότυπο Object Property. |

| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
|---|---|---|
| | 1 | Ο χρήστης επιλέγει την σύνδεση «Στιγμιότυπο Ιδιότητας Object Property» (Object Property Thing) από την παλέτα του διαγράμματος και το τοποθετεί μέσα στο διάγραμμα. |
| | 2 | Ο χρήστης επιχειρεί να συνδέσει δύο Στιγμιότυπα Κλάσης (Class Thing) μέσα στο διάγραμμα. |
| | 3 | Το σύστημα εμφανίζει ένα παράθυρο που προτρέπει τον χρήστη να ορίσει την Ιδιότητα Object Property της οποίας αποτελεί στιγμιότυπο. |
| | 4 | Ο χρήστης συμπληρώνει τα απαραίτητα στοιχεία. |
| | 5 | Η σύνδεση εμφανίζεται μέσα στο διάγραμμα. |
| | 6 | Ενημερώνεται η μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) για την προσθήκη του στοιχείου. |
| | 7 | Το σύστημα ενημερώνει την ένδειξη ανάγκης αποθήκευσης του διαγράμματος (εμφανίζεται ένας αστερίσκος στο επάνω αριστερό τμήμα του διαγράμματος, δηλαδή χρειάζεται δηλαδή σώσιμο). |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 4α | Ο χρήστης ακυρώνει την λειτουργία. |

**Περίπτωση Χρήσης 50: Διαγραφή Στοιχείου από Οντολογία.**

| **USE CASE 50** | Διαγραφή Στοιχείου από Οντολογία. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να διαγράψει κάποιο στοιχείο της οντολογίας από την οντολογία. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Επιτυγχάνεται η λειτουργία διαγραφής του στοιχείου που επιλέγεται από το χρήστη. | |
| **Failed End Condition** | Αποτυγχάνει η λειτουργία διαγραφής του στοιχείου που επιλέγεται από το χρήστη. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να διαγράψει κάποιο στοιχείο της οντολογίας από την οντολογία. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει κάποιο από τα στοιχεία της οντολογίας και επιλέγει την διαγραφή του από την οντολογία. |
| | 2 | Το σύστημα ελέγχει εάν επιτρέπεται η διαγραφή του στοιχείου από την οντολογία. Ελέγχει δηλαδή την συνέπεια της οντολογίας και φροντίζει για την αποκατάστασή της. (UC_51) |
| | 3 | Το σύστημα προχωράει στη διαγραφή του στοιχείου από το διάγραμμα (UC_52) αν περάσει από τον έλεγχο του |

| | | |
|---|---|---|
| | | προηγούμενου βήματος. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 51: Έλεγχος και Αποκατάσταση Συνέπειας.**

| **USE CASE 51** | Έλεγχος και Αποκατάσταση Συνέπειας. | |
|---|---|---|
| **Goal in Context** | Το σύστημα ελέγχει και αποκαθιστά την συνέπεια της οντολογίας σε μία διαγραφή. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας και να έχει επιλεγεί ένα στοιχείο προς διαγραφή από την οντολογία. (UC_50) | |
| **Success End Condition** | Επιτυγχάνεται ο έλεγχος και η αποκατάσταση της συνέπειας της οντολογίας. | |
| **Failed End Condition** | Καμία. | |
| **Primary, Secondary Actors** | GRAMOFONE | |
| **Trigger** | Το σύστημα ελέγχει και αποκαθιστά την συνέπεια της οντολογίας σε μία διαγραφή. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Το σύστημα ελέγχει αν το επιλεγμένο προς διαγραφή στοιχείο μπορεί να διαγραφεί από την οντολογία. |
| | 2 | Το σύστημα πραγματοποιεί όλες εκείνες τις ενέργειες που πρέπει να πραγματοποιηθούν ώστε με τη διαγραφή του στοιχείου από την οντολογία να διατηρηθεί η συνέπεια της οντολογίας. Για παράδειγμα όταν διαγράφεται ένα Datatype Property από την οντολογία τότε διαγράφεται τόσο από τις Κλάσεις οι οποίες είναι Domain για το Datatype Property όσο και από τις Κλάσεις που το κληρονομούν μέσω των Domain Κλάσεων. |
| | 3 | Το στοιχείο διαγράφεται από την οντολογία. |
| **Extensions** | **Step** | **Branching Action** |
| | 1α | Το στοιχείο δεν μπορεί να διαγραφεί από την οντολογία. Για παράδειγμα ένα Datatype Property δεν μπορεί να διαγραφεί αν έχει τεθεί κάποιος Περιορισμός πάνω σε αυτή την Ιδιότητα. |
| | | $1^{\alpha}$1. Το στοιχείο δεν διαγράφεται. |
| **Sub-Variations** | | **Branching Action** |
| | | |

**Περίπτωση Χρήσης 52: Διαγραφή Στοιχείου από Διάγραμμα.**

| | | |
|---|---|---|
| **USE CASE 52** | Διαγραφή Στοιχείου από Διάγραμμα. | |
| **Goal in Context** | Διαγραφή ενός στοιχείου από κάποιο διάγραμμα. Η διαγραφή πραγματοποιείται από το χρήστη ή το σύστημα. | |
| **Scope & Level** | GRAMOFONE, Sub-function | |
| **Preconditions** | Πρέπει να υπάρχει ενεργή κάποια οντολογία στην επιφάνεια εργασίας. | |
| **Success End Condition** | Επιτυγχάνεται η λειτουργία διαγραφής του στοιχείου από το διάγραμμα. | |
| **Failed End Condition** | Αποτυγχάνει η λειτουργία διαγραφής του στοιχείου από το διάγραμμα. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης, GRAMOFONE,<br>GRAMOFONE | |
| **Trigger** | Επιχειρείται η διαγραφή ενός στοιχείου της οντολογίας από κάποιο διάγραμμα. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Επιχειρείται η διαγραφή ενός στοιχείου της οντολογίας από κάποιο διάγραμμα. |
| | 2 | Το σύστημα ελέγχει εάν επιτρέπεται η διαγραφή του στοιχείου από το διάγραμμα. |
| | 3 | Το σύστημα διαγράφει το στοιχείο από το διάγραμμα. |
| **Extensions** | **Step** | **Branching Action** |
| | 2α | Δεν επιτρέπεται η διαγραφή του στοιχείου από το διάγραμμα. Για παράδειγμα όταν κάποιο Datatype Property περιλαμβάνεται μόνο σε μία Κλάση, τότε δεν επιτρέπεται η διαγραφή του από το διάγραμμα. |
| | | 2$^{\alpha}$1. Η λειτουργία ακυρώνεται και ο χρήστης ενημερώνεται μέσω κατάλληλου μηνύματος. |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Η διαγραφή επιλέγεται από το χρήστη. |
| | 1β | Η διαγραφή επιλέγεται από το σύστημα. |

**Περίπτωση Χρήσης 53: Τοποθέτηση Υπάρχοντος Στοιχείου σε Διάγραμμα.**

| | |
|---|---|
| **USE CASE 53** | Τοποθέτηση Υπάρχοντος Στοιχείου σε Διάγραμμα. |
| **Goal in Context** | Ο χρήστης επιθυμεί να τοποθετήσει ένα υπάρχον στοιχείο της οντολογίας σε κάποιο διάγραμμα. |
| **Scope & Level** | GRAMOFONE, Primary Task |
| **Preconditions** | Πρέπει να υπάρχει ανακτημένο στην επιφάνεια εργασίας τουλάχιστον ένα διάγραμμα. (UC_12) |
| **Success End Condition** | Η τοποθέτηση του στοιχείου επιτυγχάνεται. |
| **Failed End Condition** | Δεν πραγματοποιείται η τοποθέτηση του στοιχείου. |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE |
| **Trigger** | Ο χρήστης επιχειρεί να τοποθετήσει ένα υπάρχον στοιχείο της οντολογίας σε κάποιο διάγραμμα. |

| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
|---|---|---|
| | 1 | Ο χρήστης επιλέγει την μέθοδο με την οποία επιθυμεί να εκπληρωθεί η λειτουργία. |
| | 2 | Η λειτουργία ολοκληρώνεται. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 1α | Ο χρήστης επιλέγει από την μονάδα προβολής του περιεχομένου της οντολογίας (Model Explorer) κάποιο στοιχείο και μέσω Drag And Drop (DND) το μεταφέρει μέσα στο διάγραμμα. |
| | | 1ᵅ1. Το σύστημα ελέγχει εάν υπάρχει ήδη το στοιχείο στο διάγραμμα. |
| | | 1ᵅ1α. Το στοιχείο υπάρχει ήδη, οπότε το σύστημα δεν επιτρέπει την επαναχρησιμοποίηση του. |
| | | 1ᵅ1β. Το στοιχείο δεν υπάρχει στο διάγραμμα, οπότε προστίθεται και το διάγραμμα αποκτά την ένδειξη ανάγκης αποθήκευσης (αστερίσκος στο αριστερό άκρο του διαγράμματος). |
| | 1β | Ο χρήστης μπορεί να μεταφέρει Ιδιότητες τύπου Datatype Property μεταξύ Κλάσεων και Plain Literals μεταξύ Απαριθμήσεων (Enumerations). Η μεταφορά (copy) γίνεται μέσω Drag And Drop (DND). |
| | | 1β1. Ενημερώνεται τόσο η γραφική αναπαράσταση του στοιχείου όσο και η μονάδα προβολής του περιεχομένου της οντολογίας (Property Sheet). |

**Περίπτωση Χρήσης 54: Επεξεργασία Χαρακτηριστικών Στοιχείου.**

| **USE CASE 54** | Επεξεργασία Χαρακτηριστικών Στοιχείου. | |
|---|---|---|
| **Goal in Context** | Ο χρήστης επιθυμεί να επεξεργαστεί τα χαρακτηριστικά ενός στοιχείου της οντολογίας όπως αυτά ορίζονται από το ODM. | |
| **Scope & Level** | GRAMOFONE, Primary Task | |
| **Preconditions** | Πρέπει στο περιβάλλον εργασίας (User Interface) να υπάρχει μία ενεργή οντολογία. | |
| **Success End Condition** | Τα χαρακτηριστικά του στοιχείου της οντολογίας επεξεργάζονται με επιτυχία. | |
| **Failed End Condition** | Τα χαρακτηριστικά του στοιχείου της οντολογίας δεν επεξεργάζονται. Το GRAMOFONE παραμένει σε έγκυρη (valid) κατάσταση. | |
| **Primary, Secondary Actors** | Χρήστης,<br>GRAMOFONE | |
| **Trigger** | Ο χρήστης επιχειρεί να τροποποιήσει κάποιο από τα χαρακτηριστικά ενός στοιχείου της οντολογίας που επεξεργάζεται. | |
| **MAIN SUCCESS SCENARIO** | **Step** | **Action** |
| | 1 | Ο χρήστης επιλέγει να του παρουσιαστούν τα χαρακτηριστικά ενός στοιχείου της οντολογίας. |
| | 2 | Ο χρήστης τροποποιεί τα χαρακτηριστικά του στοιχείου της |

| | | |
|---|---|---|
| | | οντολογίας που επιθυμεί. |
| **Extensions** | **Step** | **Branching Action** |
| | | |
| **Sub-Variations** | | **Branching Action** |
| | 2α | Ο χρήστης επέλεξε Κλάση. |
| | | 2α1. Τα χαρακτηριστικά της Κλάσης της οντολογίας που μπορούν να τροποποιηθούν είναι τα παρακάτω:<br>Το όνομα της Κλάσης, οι έννοιες "oneOf", "unionOf", "intersectionOf", "complementOf", "isComplete", "isDeprecated", "equivalentWith", "disjointWith", η προσθήκη μίας Ιδιότητας τύπου Datatype Property και η προσθήκη ενός Σχολίου. Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. Επιπλέον μπορούν να επεξεργαστούν και οι Περιορισμοί που μπορεί να έχουν δημιουργηθεί στην Κλάση. |
| | 2β | Ο χρήστης επέλεξε Απαρίθμηση. |
| | | 2β1. Τα χαρακτηριστικά της Απαρίθμησης (Enumeration) της οντολογίας που μπορούν να τροποποιηθούν είναι τα παρακάτω:<br>Το όνομα της Απαρίθμησης και η προσθήκη ενός ή περισσοτέρων "Plain Literals". Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |
| | 2γ | Ο χρήστης επέλεξε Στιγμιότυπο Κλάσης. |
| | | 2γ1. Τα χαρακτηριστικά του Στιγμιότυπου Κλάσης (Class Thing) της οντολογίας που μπορούν να τροποποιηθούν είναι τα παρακάτω:<br>Το όνομα του Στιγμιότυπου Κλάσης, η προσθήκη ενός Στιγμιότυπου μίας Ιδιότητας τύπου Datatype Property (Datatype Property Thing), η προσθήκη μίας Κλάσης σαν τύπος (hasType) του Στιγμιότυπου, η προσθήκη ενός Σχολίου, και η επεξεργασία των εννοιών "differentFrom" και "sameAs". Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |
| | 2δ | Ο χρήστης επέλεξε Object Property. |
| | | 2δ1. Τα χαρακτηριστικά της Ιδιότητας Object Property της οντολογίας που μπορούν να τροποποιηθούν είναι τα παρακάτω:<br>Το όνομα της Ιδιότητας Object Property, η έννοια "inverseOf", η προσθήκη υπέρ-ιδιοτήτων, η προσθήκη ισοδύναμων ιδιοτήτων, η προσθήκη Σχολίων και η επεξεργασία των εννοιών "isTransitive", "isSymmetric", "isFunctional", "isInverseFunctional" και "isDeprecated". Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |
| | 2ε | Ο χρήστης επέλεξε Σχόλιο τύπου Comment. |
| | | 2ε1. Το χαρακτηριστικό του Σχολίου (Comment) της οντολογίας που μπορεί να τροποποιηθεί είναι το κείμενο του Σχολίου. Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |
| | 2ζ | Ο χρήστης επέλεξε Datatype Property. |
| | | 2ζ1. Τα χαρακτηριστικό της Ιδιότητας Datatype Property της οντολογίας που μπορούν να τροποποιηθούν είναι το όνομα της Ιδιότητας, το εύρος τιμών της (range), ο ορισμός υπέρ-ιδιοτήτων, ο ορισμός ισοδύναμων ιδιοτήτων, ο προσθήκη Σχολίων και η επεξεργασία των εννοιών "isFunctional" και |

| | | |
|---|---|---|
| | | "isDeprecated". Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |
| | 2η | Ο χρήστης επέλεξε Στιγμιότυπο Datatype Property. |
| | | 2η1. Το χαρακτηριστικό του Στιγμιότυπου μίας Ιδιότητας Datatype Property της οντολογίας που μπορεί να τροποποιηθεί είναι η έννοια "range" του Στιγμιότυπου. Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |
| | 2θ | Ο χρήστης επέλεξε Plain Literal. |
| | | 2θ1. Το χαρακτηριστικό του Plain Literal της οντολογίας που μπορεί να τροποποιηθεί είναι η έννοια "lexical form". Οι έννοιες αυτές ορίζονται στο ODM και έχουν ήδη περιγραφεί στο κεφάλαιο 4. |

**Ανακεφαλαίωση**

Στο κεφάλαιο περιγράφηκαν οι λειτουργικές πτυχές του GRAMOFONE μέσω της μεθοδολογίας των περιπτώσεων χρήσης (Use Cases). Οι περιπτώσεις χρήσης είναι μία software-engineering τεχνολογία η οποία παρέχει μία τυπική (formal) περιγραφή της λειτουργικότητας ενός συστήματος, η οποία είναι πλήρως κατανοητή σε οποιονδήποτε επιθυμήσει να την διαβάσει. Η φόρμα (template) περιγραφής των περιπτώσεων χρήσης του GRAMOFONE η οποία χρησιμοποιείται σε αυτό το κεφάλαιο είναι η ευρέως γνωστή φόρμα περιγραφής περιπτώσεων χρήσης που ορίζεται από τον Alistair Cockburn στο βιβλίο του «Writing Effective Use Cases». [21] Επιπλέον στο κεφάλαιο δόθηκαν δύο διαγράμματα με τις διαγραμματικές αναπαραστάσεις των περιπτώσεων χρήσης και ένας συγκεντρωτικός πίνακας των περιπτώσεων χρήσης.

Κεφάλαιο 6

# ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

**Εισαγωγή**

Στο κεφάλαιο αυτό θα παρουσιαστεί η αρχιτεκτονική του GRAMOFONE και θα περιγραφούν διεξοδικά οι διάφορες υπό-μονάδες της. Είναι σημαντικό να κατανοηθεί ότι η αρχιτεκτονική του εργαλείου GRAMOFONE είναι άρρηκτα συνδεδεμένη με τη φιλοσοφία και αρχιτεκτονική της πλατφόρμας Eclipse. Συνεπώς, στην ενότητα αυτή θα αναφέρονται και στοιχεία τα οποία δεν έχουν υλοποιηθεί στο πλαίσιο της παρούσης διπλωματικής, αλλά αποτελούν αναπόσπαστο κομμάτι του εργαλείου.

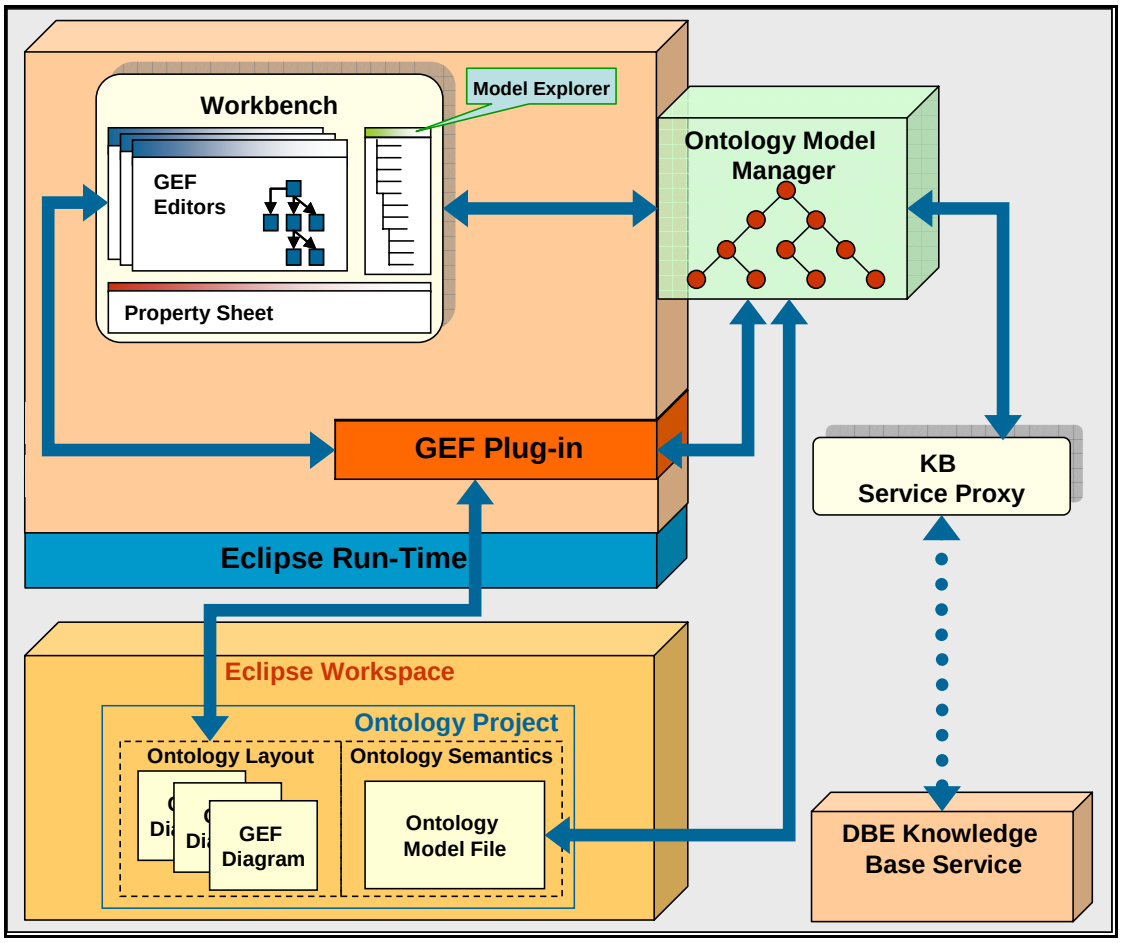
**Γενική Αρχιτεκτονική**

Όπως έχει ειπωθεί, το εργαλείο GRAMOFONE υλοποιήθηκε ως ένα επιπρόσθετο (plug-in) της πλατφόρμας Eclipse. Για την υλοποίηση του γραφικού μέρους του εργαλείου χρησιμοποιήθηκε ένα άλλο επιπρόσθετο (GEF plug-in) το οποίο παρέχει βασικές υποδομές δημιουργίας και διαχείρισης γραφικών στοιχείων.

Στο σχήμα 26 φαίνεται η γενική αρχιτεκτονική του εργαλείου με όλα του τα συστατικά μέρη καθώς και την επικοινωνία μεταξύ τους. Σε γενικές γραμμές τα συστατικά μέρη του εργαλείου μπορούν να ομαδοποιηθούν σε τέσσερις μεγάλες δομικές μονάδες. Οι μονάδες αυτές είναι:

- Η Πλατφόρμα Eclipse και τα επιπρόσθετα τα οποία χρησιμοποιούνται για να παράσχουν βασικές υποδομές όπως το γενικό περιβάλλον εργασίας (workbench), τη διαχείριση των γραφικών αντικειμένων μέσα τους καμβάδες (διαγράμματα), τα μενού βασικών επιλογών, βοήθεια, επισκόπηση των μοντέλων (οντολογιών) που δημιουργεί ο χρήστης, κλπ.
- Το τοπικό σύστημα αρχείων το οποίο παρέχει το βασικό χώρο εργασίας (workspace) όπου αποθηκεύονται μόνιμα τα έργα (projects) που δημιουργεί ο χρήστης με τη χρήση του εργαλείου.

- Το υπό-σύστημα διαχείρισης μοντέλων (οντολογιών) όπου και διατηρούνται οι βασικές δομές περιγραφής του μοντέλου (οντολογίας) που ο χρήστης δημιουργεί.
- Η υπηρεσία πρόσβασης στη Βάση Γνώσης του DBE η οποία παρέχει ένα σύνολο διεπαφών προγραμματισμού (API) βασισμένες στο πρότυπο JMI μέσω των οποίων οι οντολογίες που δημιουργεί ο χρήστης μπορούν να αποθηκευτούν ή να ανακτηθούν από τη Βάση Γνώσης του DBE.




**Εικόνα 26: Αρχιτεκτονική του GRAMOFONE**

Το εργαλείο GRAMOFONE περιλαμβάνει τις υπό-μονάδες GEF-Editors, Model Explorer και Property Sheet οι οποίες σχηματίζουν την επιφάνεια εργασίας του εργαλείου (workbench). Επιπλέον περιλαμβάνει την υπό-μονάδα Eclipse Workspace και την υπό-μονάδα Ontology Model Manager. Οι υπόλοιπες υπό-μονάδες που περιλαμβάνονται στην αρχιτεκτονική συμμετέχουν στη λειτουργία του εργαλείου. Στις επόμενες ενότητες παρατίθεται μια εκτενής περιγραφή όλων των συστατικών μερών της αρχιτεκτονικής που φαίνεται στο σχήμα 26, των λειτουργιών που το κάθε ένα διεκπεραιώνει καθώς και την αλληλεπίδρασή τους με τα υπόλοιπα συστατικά μέρη του συστήματος.

**Πυρήνας Πλατφόρμας Eclipse (Eclipse Platform Runtime)**

Όπως αναλύθηκε στο 3ο Κεφάλαιο το Eclipse Platform Runtime αποτελεί τον πυρήνα της πλατφόρμας Eclipse οπότε αναπόφευκτα αποτελεί και τμήμα της αρχιτεκτονικής του GRAMOFONE. Η λειτουργικότητα που παρέχει το Eclipse Platform Runtime όσον αφορά το GRAMOFONE είναι ίδια με αυτή που προσφέρει σε όλα τα επιπρόσθετα (plug-ins) που είναι εγκατεστημένα στην πλατφόρμα Eclipse. Κατά συνέπεια, κατά την εκκίνηση της πλατφόρμας το Platform Runtime «ανακαλύπτει» το GRAMOFONE (εφόσον έχει εγκατασταθεί στην πλατφόρμα Eclipse), διαβάζει το αρχείο-μανιφέστο του (το αρχείο "plugin.xml"), και προσθέτει το GRAMOFONE στον ειδικό κατάλογο (registry) στην κύρια μνήμη, ο οποίος χρησιμοποιείται για την καταγραφή των επιπρόσθετων (plug-ins) που είναι εγκατεστημένα στην πλατφόρμα Eclipse. Μετά το πέρας της εκκίνησης της πλατφόρμας και της δημιουργίας του καταλόγου (registry) που προαναφέρθηκε το Eclipse Platform Runtime δεν συμμετέχει στην λειτουργία του GRAMOFONE.

**GEF Plug-In**

Το πλαίσιο εργασίας GEF όπως αναφέρθηκε στο 3ο κεφάλαιο παρέχει το βασικό πλαίσιο στο οποίο μπορεί να υλοποιηθεί ένα γραφικό εργαλείο. Το GRAMOFONE, το οποίο υλοποιήθηκε στο πλαίσιο του GEF, χρησιμοποιεί σε βάθος και εκμεταλλεύεται τις περισσότερες δυνατότητες που παρέχει το συγκεκριμένο πλαίσιο εργασίας. Γίνεται λοιπόν κατανοητό πως η ύπαρξη του GEF plug-in στην πλατφόρμα Eclipse που εγκαθίσταται και το GRAMOFONE κρίνεται υποχρεωτική. Επιπλέον σε πολλές περιπτώσεις λειτουργιών του GRAMOFONE απαιτείται η χρησιμοποίηση του GEF plug-in για την αποπεράτωση των λειτουργιών αυτών. Οι λειτουργίες που παρέχονται από το GEF παρουσιάστηκαν στο κεφάλαιο 3. Κατά συνέπεια το GEF plug-in αποτελεί αναπόσπαστο κομμάτι της αρχιτεκτονικής του GRAMOFONE.

**Επιφάνεια Εργασίας Πλατφόρμας Eclipse (Eclipse Workbench)**

Το συστατικό αυτό μέρος αποτελεί την επιφάνεια εργασίας (User Interface) της πλατφόρμας Eclipse. Είναι δηλαδή το σύνολο των γραφικών στοιχείων της πλατφόρμας με τα οποία έρχεται σε επαφή ο χρήστης κατά τη διάρκεια της εργασίας του με την πλατφόρμα. Όπως είναι λογικό το γραφικό περιβάλλον εργασίας (GUI) του GRAMOFONE στηρίζεται σε αυτό της πλατφόρμας Eclipse. Το περιεχόμενο φυσικά του γραφικού αυτού περιβάλλοντος διαφέρει και ορίζεται ρητά από το GRAMOFONE, δηλαδή το GRAMOFONE ορίζει δικές του περιοχές επισκόπησης (views) και επεξεργασίας (editors), δικά του μενού (menu) και προσθέτει δικές του επιλογές στην

γραμμή εργαλείων (tool bar) της πλατφόρμας Eclipse. Ο καθορισμός των στοιχείων του γραφικού περιβάλλοντος του GRAMOFONE γίνεται μέσω του ορισμού μιας συγκεκριμένης όψης (perspective) της επιφάνειας εργασίας, δηλαδή ενός συνόλου από συγκεκριμένες περιοχές επισκόπησης (views) και περιοχές επεξεργασίας (editors) οι οποίες τοποθετούνται σε συγκεκριμένες θέσεις πάνω στην επιφάνεια εργασίας του εργαλείου.

Η επιφάνεια εργασίας του GRAMOFONE αποτελείται από τρεις βασικές υπό-μονάδες όπου η κάθε μια εξυπηρετεί και ένα διαφορετικό σκοπό. Οι υπό-μονάδες αυτές είναι α) οι Γραφικοί Επεξεργαστές Διαγραμμάτων (GEF Editors), β) Το Σύστημα Επισκόπησης του Μοντέλου (Model Explorer), και γ) ο Πίνακας Ιδιοτήτων (Property Sheet). Και τα τρία αυτά δομικά στοιχεία υλοποιήθηκαν με την χρήση των SWT (Standard Widget Toolkit) και JFace toolkits (κεφάλαιο 3).

**Γραφικοί Επεξεργαστές Διαγραμμάτων (GEF Editors)**

Μια οντολογία μπορεί να αναπτυχθεί σε πολλά διαφορετικά διαγράμματα. Κάθε τέτοιο διάγραμμα αποτελεί, από πλευράς της GEF αρχιτεκτονικής, ένα διαφορετικό γραφικό επεξεργαστή ο οποίος περιέχει στοιχεία τα οποία μπορούν να επεξεργαστούν και να τροποποιηθούν. Σύμφωνα με την GEF αρχιτεκτονική κάθε γραφικός επεξεργαστής διαγραμμάτων (GEF Editor / διάγραμμα) έχει την δική του ξεχωριστή υπόσταση και λειτουργεί ανεξάρτητα από τους άλλους γραφικούς επεξεργαστές διαγραμμάτων, δηλαδή ο ένας γραφικός επεξεργαστής διαγραμμάτων δεν γνωρίζει την ύπαρξη άλλων όμοιων επεξεργαστών ούτε βρίσκεται σε επικοινωνία μαζί τους. Για την υποστήριξη των αναγκών του GRAMOFONE επιτεύχθηκε η "ενοποίηση" αυτών των ξεχωριστών δομικών στοιχείων και η μεταξύ τους συνεργασία και επικοινωνία έτσι ώστε το ίδιο εννοιολογικό μοντέλο (οντολογία) του χρήστη να μπορεί να εμφανίζεται σε πολλά διαφορετικά διαγράμματα και να είναι δυνατή η επεξεργασία του μέσω αυτών. Όταν για παράδειγμα λαμβάνει χώρα κάποια μεταβολή σε ένα από τα διαγράμματα μίας οντολογίας, κάθε διάγραμμα, που ανήκει στην οντολογία, ενημερώνεται για την αλλαγή. Την επικοινωνία μεταξύ των διαγραμμάτων την αναλαμβάνει η υπό-μονάδα Διαχείρισης Μοντέλων (Οντολογιών) (Ontology Model Manager).

Σύμφωνα με τα παραπάνω λοιπόν στο πλαίσιο του GRAMOFONE υλοποιήθηκε η υπό-μονάδα των γραφικών επεξεργαστών διαγραμμάτων (GEF Editors) η οποία έχει σαν αποκλειστικό σκοπό την αναπαράσταση και επεξεργασία των διαγραμμάτων της εκάστοτε οντολογίας. Συμπερασματικά λοιπόν γίνεται κατανοητό πως η δημιουργία μίας οντολογίας μπορεί να γίνει σε διάφορα διαγράμματα τα οποία μάλιστα μπορούν να είναι ανοιγμένα

συγχρόνως μέσα στο περιβάλλον εργασίας της πλατφόρμας Eclipse, ωστόσο μόνο ένα μπορεί να είναι ενεργό (μπορεί δηλαδή να υποστεί επεξεργασία) κάθε φορά.

**Σύστημα Επισκόπησης του Μοντέλου (Model Explorer)**

Το Σύστημα Επισκόπησης του Μοντέλου όπως υποδηλώνει και ο τίτλος του είναι η υπό-μονάδα που είναι υπεύθυνη για την επισκόπηση του εννοιολογικού μοντέλου (οντολογίας) που δημιουργεί ο χρήστης. Το Σύστημα Επισκόπησης του Μοντέλου έχει αρκετές ακόμα λειτουργίες πέρα από την απλή αναπαράσταση των οντολογιών, λειτουργίες όπως η αναζήτηση όρων της οντολογίας και η διαγραφή όρων της οντολογίας οι οποίες θα παρουσιαστούν διεξοδικά στο επόμενο κεφάλαιο. Η αναπαράσταση των οντολογιών στο Σύστημα Επισκόπησης του Μοντέλου πραγματοποιείται μέσω της χρήσης ενός JFace δενδρικού μηχανισμού επισκόπησης (TreeViewer) (κεφάλαιο 3). Η αναπαράσταση γίνεται κατά συνέπεια σε δενδρική μορφή τα στοιχεία του οποίου (του δένδρου) ορίζονται ρητά από το GRAMOFONE με βάση το ODM (Ontology Definition Metamodel) μετά-μοντέλο. Στο σημείο αυτό πρέπει να σημειωθεί πως μία οντολογία, όπως ορίζεται από το ODM, δεν είναι δενδροειδής σχηματισμός (π.χ. μία κλάση μπορεί να έχει δύο υπέρ-κλάσεις). Για τον λόγο αυτό αποφασίστηκε να ακολουθηθεί η ίδια πολιτική που ακολουθείται από άλλους δημοφιλείς UML editors (π.χ. Poseidon). Σύμφωνα με την πολιτική αυτή τα στοιχεία της οντολογίας (κλάσεις, απαριθμήσεις κλπ.) τοποθετούνται σαν κόμβοι του δένδρου σε πρώτο επίπεδο και στο επόμενο επίπεδο τοποθετούνται τόσο τα χαρακτηριστικά των στοιχείων αυτών (π.χ. ιδιότητες τύπου datatype property) όσο και οι σχέσεις των στοιχείων αυτών με άλλα στοιχεία της οντολογίας (π.χ. υπό-κλάσεις).

**Πίνακας Ιδιοτήτων (Property Sheet)**

Η υπό-μονάδα του Πίνακα Ιδιοτήτων είναι υπεύθυνη για την παρουσίαση των τιμών των ιδιοτήτων (χαρακτηριστικών) των στοιχείων που απαρτίζουν την εκάστοτε οντολογία. Οι ιδιότητες αυτές ορίζονται ρητά στο μετά-μοντέλο ODM. Τέτοιες ιδιότητες είναι για παράδειγμα τα ονόματα και οι υπό-κλάσεις των ODM Κλάσεων, τα ονόματα των Στιγμιότυπων των κλάσεων (Class Thing) κλπ. Το πληροφοριακό περιεχόμενο της υπό-μονάδας αυτής ενημερώνεται και ανανεώνεται κατά την επιλογή κάποιου στοιχείου είτε σε κάποιο ενεργό διάγραμμα είτε στο Σύστημα Επισκόπησης του Μοντέλου (Model Explorer). Για παράδειγμα αν επιλεγεί κάποια κλάση της οντολογίας πάνω σε κάποιο διάγραμμα που την περιέχει, το περιεχόμενο του Πίνακα Ιδιοτήτων ανανεώνεται και παρουσιάζει τις ιδιότητες της κλάσης αυτής και τις τιμές που έχουν οι ιδιότητες αυτές.

Επιπλέον είναι δυνατή η τροποποίηση των στοιχείων της εκάστοτε οντολογίας μέσω των φορμών (widgets) που συγκροτούν τον Πίνακα Ιδιοτήτων

Οι τρεις υπό-μονάδες που περιγράφηκαν στην ενότητα αυτή υποστηρίζουν στην ουσία το περιβάλλον εργασίας (workbench) του εργαλείου GRAMOFONE. Η περιγραφή έγινε με όρους λειτουργιών που οι υπό-μονάδες αυτές επιτελούν. Η επικοινωνία των υπό-μονάδων με άλλα δομικά στοιχεία του συστήματος θα περιγραφεί μετά την ολοκλήρωση της περιγραφής όλων των δομικών στοιχείων της αρχιτεκτονικής του GRAMOFONE. Το πώς αυτές οι υπό-μονάδες υλοποιούνται από πλευράς διεπαφών χρήστη (User Interface) θα αναλυθεί στο επόμενο κεφάλαιο.

**Χώρος Εργασίας (Workspace)**

Όπως παρουσιάστηκε στο 3ο κεφάλαιο ο χώρος εργασίας (workspace), στο πλαίσιο της πλατφόρμας Eclipse, αποτελεί ουσιαστικά τον αποθηκευτικό χώρο στο τοπικό σύστημα αρχείων στον οποίο αποθηκεύονται τα έργα (projects) που δημιουργούνται από τους χρήστες μέσα από τη χρήση της πλατφόρμας, καθώς επίσης και τα αρχεία που δημιουργούνται στο πλαίσιο κάθε έργου (project).

Στο πλαίσιο του GRAMOFONE η υπό-μονάδα του χώρου εργασίας (workspace) αποθηκεύει τις οντολογίες που δημιουργούνται μέσω της χρήσης του GRAMOFONE. Πιο συγκεκριμένα κάθε οντολογία που δημιουργείται αντιστοιχεί σε ένα καινούργιο έργο (project) για την πλατφόρμα Eclipse. Αυτό πρακτικά σημαίνει την ύπαρξη ενός υποκαταλόγου (folder) στο τοπικό σύστημα αρχείων. Το όνομα του υποκαταλόγου ταυτίζεται με το όνομα της οντολογίας που έχει οριστεί από τον χρήστη. Αυτός ο υποκατάλογος δημιουργείται σε συγκεκριμένο χώρο στο τοπικό σύστημα αρχείων και πιο συγκεκριμένα δημιουργείται μέσα στον υποκατάλογο "workspace" του βασικού καταλόγου εγκατάστασης της πλατφόρμας Eclipse. Αν δηλαδή η πλατφόρμα Eclipse είναι εγκατεστημένη στο «c:\», τότε ο χώρος εργασίας (workspace) του GRAMOFONE θα είναι ο υποκατάλογος «c:\eclipse\workspace». Μέσα σε αυτόν τον υποκατάλογο θα αποθηκεύονται οι οντολογίες με την μορφή έργων (projects). Οτιδήποτε δημιουργείται στο πλαίσιο της οντολογίας αποθηκεύεται μέσα σε αυτόν τον ειδικά διαμορφωμένο υποκατάλογο.

Το πρώτο αρχείο που δημιουργείται αυτόματα και ταυτόχρονα με τη δημιουργία μίας νέας οντολογίας είναι το ".project" αρχείο, το οποίο είναι ένα xml έγγραφο που περιέχει κάποιες πληροφορίες και ιδιότητες σχετικές με το έργο (project) που δημιουργήθηκε και λειτουργεί ουσιαστικά σαν αναγνωριστικό, για την πλατφόρμα Eclipse, αρχείο του γεγονότος πως ο

υποκατάλογος που το περιέχει αποτελεί κάποιο έργο (project) για την πλατφόρμα Eclipse. Ο χρήστης δεν χρειάζεται και δεν επεμβαίνει καθόλου σε αυτό το αρχείο σε όλη την διάρκεια ορισμού της οντολογίας.

Στον υποκατάλογο κάθε οντολογίας αποθηκεύονται ακόμα δύο είδη αρχείων. Το αρχείο στο οποίο αποθηκεύονται οι κατάλληλες δομές οι οποίες περιγράφουν τη σημασιολογία (semantics) της οντολογίας που αναπτύσσει ο χρήστης και τα διαγράμματα που δημιουργούνται από τον χρήστη. Τα αρχεία αυτά παρουσιάζονται παρακάτω.

**Αρχείο Αποθήκευσης Οντολογίας (Ontology Model File)**

Όπως υποδηλώνει η ονομασία του, στο Αρχείο Αποθήκευσης Οντολογίας (Ontology Model File) αποθηκεύεται η εκάστοτε οντολογία που δημιουργείται. Πρόκειται για ένα αρχείο το οποίο δημιουργείται αυτόματα και ταυτόχρονα με τη δημιουργία της οντολογίας μέσα στον υποκατάλογο της οντολογίας. Στο αρχείο αυτό αποθηκεύονται οι κατάλληλες δομές οι οποίες περιγράφουν τη σημασιολογία (semantics) της οντολογίας που αναπτύσσει ο χρήστης χωρίς επιπλέον πληροφορία σχετικά με τη γραφική αναπαράσταση των όρων της οντολογίας, κλπ. Η αποθήκευση της οντολογίας, που αναπτύσσεται από το χρήστη στο περιβάλλον εργασίας του GRAMOFONE, στο Αρχείο Αποθήκευσης Οντολογίας πραγματοποιείται μέσω της επιλογής από το χρήστη της αντίστοιχης λειτουργίας στο περιβάλλον εργασίας (User Interface).

**Γραφική Αναπαράσταση Οντολογιών (Ontology Layout)**

Η υπό-μονάδα της Γραφικής Αναπαράστασης Οντολογιών αποτελείται από τα αρχεία που αντιστοιχούν στα διαγράμματα που δημιουργεί ο χρήστης κατά τη διάρκεια ανάπτυξης μιας οντολογίας. Όπως αναφέρθηκε, κάθε διάγραμμα (GEF Editor), αποθηκεύεται στο τοπικό σύστημα αρχείων σε κάποιο αρχείο. Το αρχείο αυτό έχει επέκταση ".diagram", και όνομα αυτό του διαγράμματος στο οποίο αντιστοιχεί. Τα αρχεία αυτά είναι σε απόλυτο συγχρονισμό με τα διαγράμματα τα οποία αντιπροσωπεύουν. Το σύνολο αυτών των αρχείων συγκροτούν την υπό-μονάδα της Γραφικής Αναπαράστασης Οντολογιών.

Να σημειωθεί πως κάθε αλλαγή που πραγματοποιείται σε κάποιο διάγραμμα γραφικά δεν αποθηκεύεται αυτόματα στο αρχείο του, αλλά ο χρήστης θα πρέπει να το αποθηκεύσει ο ίδιος. Αυτό σημαίνει πως αν ο χρήστης κλείσει κάποιο διάγραμμα το οποίο περιλαμβάνει αλλαγές οι οποίες δεν έχουν αποθηκευτεί από το χρήστη, τότε εμφανίζεται ένα παράθυρο διαλόγου (dialog) που προτρέπει το χρήστη να αποθηκεύσει το διάγραμμα, και αν εκείνος δεν το αποθηκεύσει οι αλλαγές αυτές χάνονται. Κάθε φορά που σε κάποιο διάγραμμα πραγματοποιούνται αλλαγές που δεν έχουν αποθηκευτεί, τότε εμφανίζεται μία ένδειξη στο

πάνω αριστερό τμήμα του διαγράμματος που υποδηλώνει πως το διάγραμμα χρειάζεται αποθήκευση.

**Υπηρεσία Πρόσβασης στη Βάση Γνώσης του DBE (DBE Knowledge Base Service)**

Η Υπηρεσία Πρόσβασης στη Βάση Γνώσης του DBE αποτελεί μία βασική υπηρεσία για το DBE, η οποία εξάγει τμήμα της λειτουργικότητας την οποία υποστηρίζει η Βάση Γνώσης (Knowledge Base) και την παρέχει στη διάθεση του χρήστη. Η λειτουργικότητα αυτή που εξάγεται είναι απαραίτητη και χρησιμοποιείται από το GRAMOFONE. Πρόκειται για μία υπηρεσία μετά-δεδομένων βασισμένη στο JMI πρότυπο. Η Βάση Γνώσης παρουσιάστηκε και αναλύθηκε διεξοδικά στο 3ο κεφάλαιο. Εν συντομία αποτελεί για το GRAMOFONE μία βάση δεδομένων στην οποία ο χρήστης του GRAMOFONE μπορεί να αποθηκεύσει αν επιθυμεί κάποια οντολογία ή να ανακτήσει κάποια από τις οντολογίες που βρίσκονται αποθηκευμένες στη Βάση Γνώσης. Η υπηρεσία πρόσβασης στη Βάση Γνώσης του DBE παρέχει ένα σύνολο διεπαφών προγραμματισμού (API) βασισμένες στο πρότυπο JMI μέσω των οποίων οι οντολογίες που δημιουργεί ο χρήστης μπορούν να αποθηκευτούν ή να ανακτηθούν από τη Βάση Γνώσης του DBE [16].

**Εξυπηρετητής Διαμεσολάβησης της Υπηρεσίας Πρόσβασης στη Βάση Γνώσης (KB Service Proxy)**

Η υπηρεσία πρόσβασης στη Βάση Γνώσης του DBE παρέχει μηχανισμούς εξυπηρετητών διαμεσολάβησης (proxies) οι οποίοι αναλαμβάνουν την επικοινωνία μεταξύ του GRAMOFONE και της υπηρεσίας πρόσβασης στη Βάση Γνώσης του DBE. Μέσω της χρήσης αυτών των εξυπηρετητών διαμεσολάβησης παρέχονται οι παρακάτω λειτουργίες από το GRAMOFONE:

- Περιήγηση στις οντολογίες που είναι αποθηκευμένες στη Βάση Γνώσης.
- Αποθήκευση οντολογιών στη Βάση Γνώσης.
- Ανάκτηση οντολογιών από τη Βάση Γνώσης.
- Αναζήτηση οντολογιών αποθηκευμένων στη Βάση Γνώσης.
- Εξαγωγή οντολογιών σε μορφή εγγράφων XMI 1.2.

**Υπό-Μονάδα Διαχείρισης Μοντέλων (Οντολογιών) (Ontology Model Manager)**

Η υπό-μονάδα Διαχείρισης Οντολογιών (Ontology Model Manager) αποτελεί ουσιαστικά την "καρδιά" του GRAMOFONE. Όπως φαίνεται και στην εικόνα 26 σχεδόν όλες οι αλληλεπιδράσεις μεταξύ των υπό-μονάδων του GRAMOFONE περνούν από την υπό-μονάδα Διαχείρισης Οντολογιών. Η υπό-μονάδα Διαχείρισης Οντολογιών είναι υπεύθυνη

για τη διαχείριση της οντολογίας που βρίσκεται υπό επεξεργασία. Οποιαδήποτε λειτουργία έχει σχέση με την επεξεργασία της οντολογίας περνάει από αυτή την υπό-μονάδα. Η χρησιμότητα και η λειτουργικότητά της θα φανεί ακόμα περισσότερο στην περιγραφή των αλληλεπιδράσεων μεταξύ των υπό-μονάδων στη συνέχεια.

Επιπλέον η υπό-μονάδα Διαχείρισης Οντολογιών έχει τον ρόλο της κύριας μνήμης του GRAMOFONE καθώς διατηρεί τις κατάλληλες δομές οι οποίες περιγράφουν τη σημασιολογία (semantics) της οντολογίας που αναπτύσσει ο χρήστης στο γραφικό περιβάλλον εργασίας (GUI), δηλαδή η υπό-μονάδα Διαχείρισης Οντολογιών διατηρεί την υπό-επεξεργασία οντολογία. Η σημασιολογία (semantics) της οντολογίας που αναπτύσσει ο χρήστης στο γραφικό περιβάλλον εργασίας (GUI) απεικονίζεται από την υπό-μονάδα του Συστήματος Επισκόπησης του Μοντέλου (Model Explorer) που περιγράφηκε προηγουμένως, δηλαδή αποτελεί την είσοδο (input) του JFace δενδρικού μηχανισμού επισκόπησης (TreeViewer) της υπό-μονάδας αυτής. Επιπλέον η υπό-επεξεργασία οντολογία αποθηκεύεται τόσο στη Βάση Γνώσης (Knowledge Base) όσο και στο τοπικό σύστημα αρχείων, όταν ο χρήστης το επιλέξει.

**Περιγραφή Αλληλεπιδράσεων Μεταξύ Των Υπό-Μονάδων**

Είναι προφανές πως οι υπό-μονάδες που απαρτίζουν την αρχιτεκτονική του GRAMOFONE θα έχουν κάποια μορφή επικοινωνίας και αλληλεπίδρασης μεταξύ τους. Οι δυνατές αλληλεπιδράσεις που μπορεί να υπάρξουν μεταξύ των υπό-μονάδων του GRAMOFONE θα παρουσιαστούν παρακάτω.

Η υπό-μονάδα Διαχείρισης Οντολογιών (από εδώ και στο εξής Ontology Model Manager) έχει την ιδιότητα να παρακολουθεί τις επιλογές (selection changes) των στοιχείων που γίνονται στην υπό-μονάδα των γραφικών επεξεργαστών διαγραμμάτων (από εδώ και στο εξής GEF Editors), δηλαδή στα διαγράμματα. Κάθε φορά που επιλέγεται κάποιο γραφικό στοιχείο σε ένα διάγραμμα η επιλογή αυτή στέλνεται στον Ontology Model Manager και αυτός με την σειρά του ενημερώνει την υπό-μονάδα του Συστήματος Επισκόπησης του Μοντέλου (από εδώ και στο εξής Model Explorer) έτσι ώστε ο Model Explorer, ο οποίος αποτελείται από ένα JFace δενδρικό μηχανισμό επισκόπησης (TreeViewer), να εστιάζει (highlight) στο στοιχείο του μηχανισμού (TreeViewer) το οποίο αντιστοιχεί στο γραφικό στοιχείο που επιλέχθηκε.

Επιπλέον ο Ontology Model Manager έχει την ιδιότητα να παρακολουθεί τις αλλαγές των ιδιοτήτων (property changes) των γραφικών στοιχείων των διαγραμμάτων. Κάθε φορά που κάποιο γραφικό στοιχείο ενός διαγράμματος μεταβάλλεται (αλλάζει κάποια ιδιότητά του) γραφικά (μέσω direct editing), η αλλαγή αυτή στέλνεται στον Ontology Model Manager και

αυτός με την σειρά του ενημερώνει τον Model Explorer έτσι ώστε να ανανεώσει το περιεχόμενό του ώστε να περιλαμβάνει την αλλαγή που πραγματοποιήθηκε γραφικά. Παράλληλα ενημερώνεται και η υπό-μονάδα του Αρχείου Αποθήκευσης Οντολογίας (από εδώ και στο εξής Ontology Model File) για την αλλαγή. Η ίδια διαδικασία ακολουθείται και κατά την πρόσθεση στοιχείων (κλάσεις, στιγμιότυπα κλπ.) στα διαγράμματα.

Όταν μεταβάλλεται κάποιο στοιχείο γραφικά (μέσω direct editing), ο Ontology Model Manager ενημερώνεται για την μεταβολή μέσω της GEF Plug-In υπό-μονάδας (από εδώ και στο εξής GEF Plug-In) και αναλαμβάνει την αποπεράτωση της γραφικής αναπαράστασης της μεταβολής που επιλέχθηκε από τον χρήστη (δηλαδή την γραφική απεικόνιση της μεταβολής, π.χ. την γραφική αλλαγή του ονόματος μίας ODM Κλάσης). Η λειτουργία αυτή υλοποιείται μέσω του GEF Plug-In επίσης.

Ακόμα κάνοντας διπλό κλικ σε κάποιο από τα στοιχεία του JFace δενδρικού μηχανισμού επισκόπησης (TreeViewer) του Model Explorer, το στοιχείο αυτό στέλνεται στον Ontology Model Manager και αυτός με την σειρά του ανακτά το πρώτο από τα διαγράμματα που περιέχει το επιλεγμένο στοιχείο, το ανοίγει στην επιφάνεια εργασίας και εστιάζει (focus) στο επιλεγμένο στοιχείο.

Ο Ontology Model Manager παρακολουθεί επιπλέον τα στοιχεία που επιλέγονται στον JFace δενδρικό μηχανισμό επισκόπησης (TreeViewer) του Model Explorer. Κάθε φορά που επιλέγεται κάποιο στοιχείο στον μηχανισμό (TreeViewer) στέλνεται αυτή η επιλογή στον Ontology Model Manager και εκείνος με την σειρά του ενημερώνει την υπό-μονάδα του Πίνακα Ιδιοτήτων (από εδώ και στο εξής Property Sheet) ότι πρέπει να ανανεώσει το περιεχόμενό της ώστε να παρουσιάζει τις ιδιότητες του στοιχείου που επιλέχθηκε στον μηχανισμό επισκόπησης (TreeViewer), όπως και γίνεται.

Επιπλέον όταν κάποιο στοιχείο της οντολογίας μεταβάλλεται μέσω της χρήσης του Property Sheet, ο Ontology Model Manager λαμβάνει γνώση της αλλαγής αυτής και ενημερώνει τον Model Explorer έτσι ώστε να ανανεώσει το περιεχόμενό του για να περιλαμβάνει την αλλαγή αυτή. Παράλληλα ενημερώνεται και το Ontology Model File για την αλλαγή. Ο Ontology Model Manager είναι υπεύθυνος επιπλέον να ενημερώσει και τα διαγράμματα που περιέχουν το στοιχείο που μεταβλήθηκε. Η αναζήτηση των διαγραμμάτων αυτών γίνεται σε δύο φάσεις. Στην πρώτη φάση ο Ontology Model Manager ελέγχει τα ενεργά στην επιφάνεια εργασίας διαγράμματα και αν κάποιο από αυτά περιέχει το στοιχείο που μεταβλήθηκε τότε το ενημερώνει για την μεταβολή. Η ενημέρωση γίνεται μέσω του GEF Plug-In το οποίο αναλαμβάνει να πραγματοποιήσει την ενημέρωση αυτή. Σε δεύτερη φάση ο Ontology Model Manager ελέγχει τα υπόλοιπα διαγράμματα της οντολογίας που βρίσκονται στο χώρο εργασίας (workspace). Ο έλεγχος πραγματοποιείται

μέσω του GEF Plug-In και αν κάποιο από τα διαγράμματα περιέχει το στοιχείο που μεταβλήθηκε ενημερώνεται ανάλογα μέσω του GEF Plug-In επίσης.
Η λειτουργία αποθήκευσης και ανάκτησης των διαγραμμάτων από το τοπικό σύστημα αρχείων πραγματοποιείται μέσω του GEF Plug-In.
Κατά την ανάκτηση οντολογιών αποθηκευμένων στο τοπικό σύστημα ο Ontology Model Manager ανακτά τα περιεχόμενα της οντολογίας από το Ontology Model File και τα χρησιμοποιεί σαν είσοδο στο δενδρικό μηχανισμό επισκόπησης (TreeViewer) του Model Explorer.
Όταν ο χρήστης επιθυμεί να αποθηκεύσει μία οντολογία στη Βάση Γνώσης του DBE ο Ontology Model Manager αναλαμβάνει χρησιμοποιώντας την υπό-μονάδα του Εξυπηρετητή Διαμεσολάβησης της Υπηρεσίας Πρόσβασης στη Βάση Γνώσης (από εδώ και στο εξής KB Service Proxy) να προωθήσει το αίτημα αυτό στην υπό-μονάδα της Υπηρεσίας Πρόσβασης στη Βάση Γνώσης (από εδώ και στο εξής DBE Knowledge Base Service ), η οποία με την σειρά της αναλαμβάνει να διεκπεραιώσει το αίτημα αναλαμβάνοντας την αποθήκευση της οντολογίας στη Βάση Γνώσης.
Όταν ο χρήστης επιθυμεί να ανακτήσει μία οντολογία που είναι αποθηκευμένη στη Βάση Γνώσης στέλνεται το αίτημα μέσω του Ontology Model Manager στον KB Service Proxy, ο οποίος με την σειρά του αναλαμβάνει την προώθηση του αιτήματος στην DBE Knowledge Base Service και εκείνη με την σειρά της φροντίζει την ανάκτηση της οντολογίας από τη Βάση Γνώσης και την απόδοσή της στον Ontology Model Manager. Στην συνέχεια ο Ontology Model Manager αναλαμβάνει την δημιουργία της ανακτημένης οντολογίας στην επιφάνεια εργασίας του GRAMOFONE.

Μετά την περιγραφή τόσο των υπό-μονάδων όσο και των αλληλεπιδράσεων μεταξύ των υπό-μονάδων που συγκροτούν τα στοιχεία της αρχιτεκτονικής του GRAMOFONE γίνεται πλέον ξεκάθαρο πώς προκύπτει η εικόνα 26 της αρχιτεκτονικής του GRAMOFONE.

**Ανακεφαλαίωση**

Στο κεφάλαιο αυτό παρουσιάστηκε η αρχιτεκτονική του GRAMOFONE. Αρχικά παρουσιάστηκε μία γενική εικόνα της αρχιτεκτονικής του εργαλείου, στη συνέχεια αναλύθηκαν διεξοδικά οι υπό-μονάδες της αρχιτεκτονικής και τέλος περιγράφηκαν οι αλληλεπιδράσεις μεταξύ των υπό-μονάδων αυτών.

Κεφάλαιο 7

# ΓΡΑΦΙΚΟ ΠΕΡΙΒΑΛΛΟΝ ΕΡΓΑΣΙΑΣ GRAMOFONE

**Εισαγωγή**

Στο κεφάλαιο θα παρουσιαστεί το γραφικό περιβάλλον εργασίας (GUI) του GRAMOFONE. Αρχικά θα περιγράψουμε τις γενικές αρχές σχεδιασμού [20] οι οποίες λήφθηκαν υπόψη κατά τη διάρκεια του σχεδιασμού του περιβάλλοντος εργασίας του εργαλείου και κατόπιν θα περιγράψουμε με τη βοήθεια στιγμιότυπων (screenshots) πλήρως το περιβάλλον εργασίας του εργαλείου GRAMOFONE.

**Βασικές Αρχές Σχεδιασμού (User Interface Guidelines)**

- **Ορατότητα της Κατάστασης του Συστήματος (Visibility of System Status):** Το περιβάλλον εργασίας του GRAMOFONE ενημερώνει διαρκώς το χρήστη σχετικά με το που βρίσκεται και τι κάνει. Αυτό επιτυγχάνεται με τη χρήση μίας μπάρας μηνυμάτων (status bar) που υπάρχει στο κάτω αριστερό άκρο της επιφάνειας εργασίας του GRAMOFONE η οποία εμφανίζει μηνύματα λάθους σε ορισμένες περιπτώσεις λανθασμένων γραφικών επιλογών του χρήστη (θα περιγραφεί παρακάτω). Επιπλέον η ενημέρωση του χρήστη πραγματοποιείται μέσω της χρήσης ορθών τίτλων στους διαλόγους εφαρμογής.
- **Αντιστοίχιση μεταξύ Συστήματος και Πραγματικού Κόσμου (Match Between System and the Real World):** Κατά τον σχεδιασμό του User Interface συνυπολογίστηκε το κοινό στο οποίο αυτό απευθύνεται και έτσι χρησιμοποιήθηκαν όροι οικείοι και κατανοητοί για το σύνολο των ενδιαφερομένων.
- **Έλεγχος και Ελευθερία Χρήστη (User Control and Freedom):** Ο χρήστης στις περισσότερες περιπτώσεις έχει την ελευθερία να ενεργήσει με τον τρόπο και τη σειρά που θέλει ο ίδιος, χωρίς να είναι δεσμευμένος να ακολουθήσει μια ακριβή και συγκεκριμένη σειρά ενεργειών. Μειώθηκαν οι διάλογοι που ενημερώνουν το χρήστη

μόνο στους απαραίτητους, δηλαδή, σε εκείνους στους οποίους εγκυμονεί κίνδυνος να απολεσθούν δεδομένα από κατά λάθος ενέργειες.

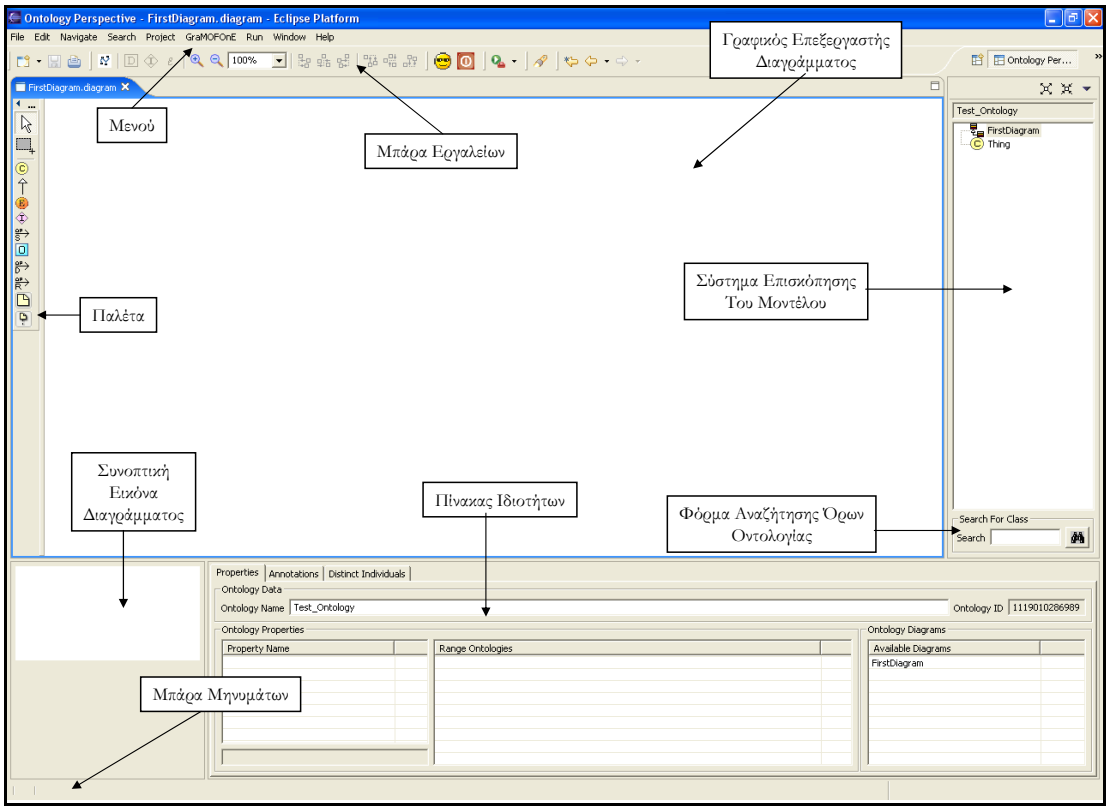
- **Συνέπεια και Πρότυπα (Consistency and Standards):** Σε όλη την εφαρμογή ακολουθείται η ίδια λογική σχεδιασμού (π.χ. ίδια χρώματα, κοινή ονομασία κουμπιών που εκτελούν την ίδια ενέργεια). Επίσης έγινε προσπάθεια ώστε η εφαρμογή να ακολουθεί τα πρότυπα άλλων ευρέως χρησιμοποιούμενων εφαρμογών ώστε να είναι όσο το δυνατόν πιο οικεία στους χρήστες.
- **Αναγνώριση παρά Μνήμη (Recognition rather than Recall):** Ο χρήστης δεν είναι υποχρεωμένος να θυμάται πληροφορία μεταξύ των ενεργειών που κάνει. Με αυτή την καθοδήγηση κατά νου, σε κάθε σημείο στην εφαρμογή υπάρχουν επαρκείς επεξηγήσεις ώστε ο χρήστης να καθοδηγείται στο πως θα κάνει αυτά που θέλει να πραγματοποιήσει. Οι επεξηγήσεις είναι σύντομα επεξηγηματικά μηνύματα (tool tips) ή ακόμα και σύντομες επιγραφές (labels) που επεξηγούν την διαδικασία που μπορεί να ακολουθηθεί.
- **Αποτροπή Λαθών (Error Prevention):** Ο χρήστης εμποδίζεται από το να κάνει λάθη με το να του προσφέρονται όλες οι δυνατές επιλογές όπου χρειάζεται ώστε ποτέ να μην ενεργεί αυθαίρετα, παράγοντας στον όποιο κατά κύριο λόγο οφείλεται η πρόκληση λαθών.
- **Ευελιξία και Αποδοτικότητα Χρήσης (Flexibility and Efficiency of Use):** Το σύστημα απευθύνεται τόσο σε έμπειρους χρήστες όσο και σε λιγότερο έμπειρους ή ακόμα και αρχάριους επιτρέποντας την ταυτόχρονη εξυπηρέτηση όλων των χρηστών, δίχως η εξυπηρέτηση ενός είδους χρηστών να εμπλέκεται στην εξυπηρέτηση των υπολοίπων. Έχει γίνει αρκετή προσπάθεια να έχει η εφαρμογή καλή απόκριση. Οι όποιες καθυστερήσεις που συμβαίνουν οφείλονται τόσο στην τεράστια ποσότητα πληροφορίας που πρέπει να φορτωθεί από τη Βάση Γνώσης (κατά την ανάκτηση οντολογιών), όσο και στις δυνατότητες του τοπικού συστήματος στο οποίο λειτουργεί η πλατφόρμα Eclipse και κατ'επέκταση το GRAMOFONE.
- **Αισθητική και Μινιμαλιστική Σχεδίαση (Aesthetic and Minimalistic Design):** Η αισθητική (Look & Feel) της εφαρμογής είναι όμοια πάντα με αυτή του λειτουργικού συστήματος, ώστε να είναι πιο κοντά στις αισθητικές προτιμήσεις του χρήστη. Εκτός αυτού, στο χρήστη δεν παρέχονται περιττές πληροφορίες και οι επεξηγήσεις παραμένουν σύντομες, διότι σε αντίθετη περίπτωση αυτό θα εμπόδιζε τον χρήστη να εστιάσει την προσοχή του στην πληροφορία. Ακόμη και στην περίπτωση των λαθών, η επισήμανση τους γίνεται με τρόπο σύντομο ώστε να φαίνεται ξεκάθαρα το μήνυμα που θέλουν να δώσουν στο χρήστη.

- **Βοήθεια Χρηστών στην Αναγνώριση, Διάγνωση και Ανάνηψη των Λαθών (Help Users Recognize, Diagnose, and Recover from Errors):** Τα μηνύματα λάθους είναι λακωνικά, εύκολα κατανοητά και απόλυτα ακριβή έτσι ώστε να τα καταλαβαίνει αμέσως ο χρήστης και να μπορεί να διορθώσει εύκολα τα λάθη του.

Στην συνέχεια θα περιγραφεί και θα αναλυθεί λεπτομερώς τόσο το γραφικό περιβάλλον της εφαρμογής όσο και η λειτουργικότητα που υποστηρίζεται από αυτό. Οι περιπτώσεις χρήσης του GRAMOFONE παρουσιάστηκαν και αναλύθηκαν λεπτομερώς στο κεφάλαιο 5. Κατά συνέπεια δεν θα επαναληφθεί η περιγραφή τους αλλά θα παρουσιαστεί με ποιο τρόπο το GRAMOFONE υποστηρίζει αυτές τις λειτουργίες.

**GRAMOFONE Perspective (Ontology Perspective)**

Στο τρίτο κεφάλαιο ορίστηκε η έννοια του «Perspective». Εν συντομία με τον όρο αυτό εννοούμε μία συγκεκριμένη σύνθεση από περιοχές επισκόπησης (views) και περιοχές επεξεργασίας (editors), των οποίων οι θέσεις πάνω στην επιφάνεια εργασίας είναι επίσης προκαθορισμένες. Στην ενότητα αυτή θα παρουσιαστεί το Perspective που ορίστηκε για τις ανάγκες του GRAMOFONE. Θα μπορούσε να ειπωθεί πως ουσιαστικά το Ontology Perspective ταυτίζεται με το γραφικό περιβάλλον του GRAMOFONE. Στην εικόνα 27 παρουσιάζεται ένα στιγμιότυπο (screenshot) του Perspective που ορίζεται από το GRAMOFONE (το οποίο ονομάζεται «Ontology Perspective»).




**Εικόνα 27: GRAMOFONE Perspective**

Όπως φαίνεται στην εικόνα 27 το Perspective που ορίζεται στο GRAMOFONE αποτελείται από τα παρακάτω τμήματα:

- Μενού (Menu): Το GRAMOFONE ορίζει ένα δικό του μενού (GraMOFOnE) το οποίο περιέχει τις βασικές λειτουργίες του εργαλείου. Διατηρούνται φυσικά τα μενού που ορίζονται από την πλατφόρμα Eclipse.
- Μπάρα Εργαλείων (Tool Bar): Στη μπάρα εργαλείων της πλατφόρμας Eclipse προστίθονται επιλογές (action buttons) για την αποπεράτωση συγκεκριμένων λειτουργιών του εργαλείου.
- Γραφικός Επεξεργαστής Διαγράμματος (GEF Editor): Όπως περιγράφηκε στο κεφάλαιο της αρχιτεκτονικής του GRAMOFONE (κεφάλαιο έξι), η υπό-μονάδα στην οποία ανακτώνται και επεξεργάζονται τα διαγράμματα κάθε οντολογίας είναι η υπό-μονάδα των Γραφικών Επεξεργαστών Διαγραμμάτων (GEF Editors) η οποία συγκροτείται από έναν ή περισσότερους Γραφικούς Επεξεργαστές Διαγραμμάτων σαν αυτόν που φαίνεται στην εικόνα 27.
- Παλέτα (Palette): Κάθε Γραφικός Επεξεργαστής Διαγράμματος αποτελείται από μία παλέτα, από την οποία ο χρήστης έχει την δυνατότητα να εισάγει στοιχεία μέσα στα διαγράμματα που επεξεργάζεται.

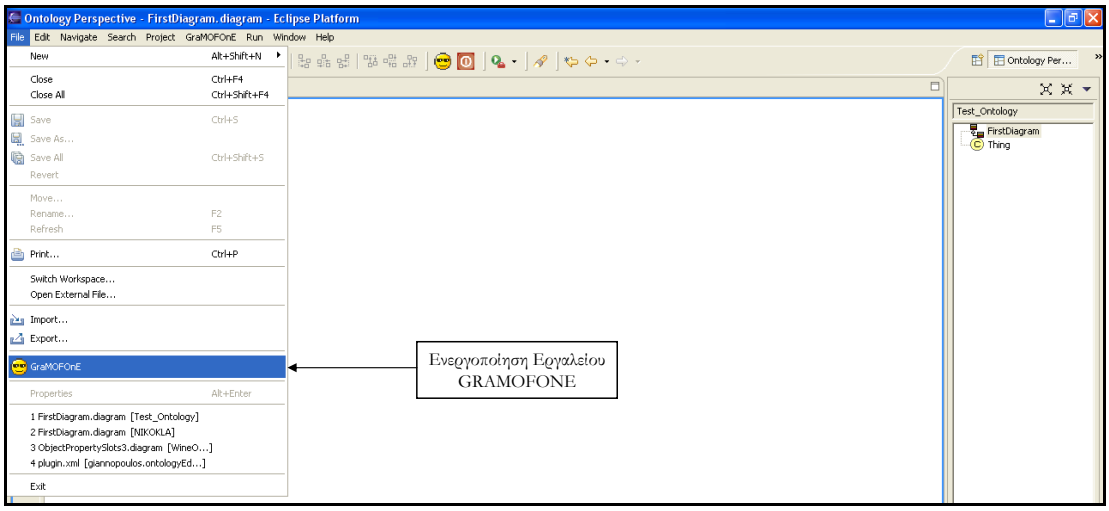
- Σύστημα Επισκόπησης του Μοντέλου (Model Explorer): Η υλοποίηση της υπό-μονάδας του Συστήματος Επισκόπησης του Μοντέλου (Model Explorer) όπως αναλύθηκε στο κεφάλαιο έξι της αρχιτεκτονικής του GRAMOFONE.
- Φόρμα Αναζήτησης Όρων Οντολογίας: Μέσω της φόρμας που περιλαμβάνεται στο Σύστημα Επισκόπησης του Μοντέλου (Model Explorer), ο χρήστης έχει την δυνατότητα να αναζητά όρους που έχει εισάγει στην οντολογία που επεξεργάζεται. Τα αποτελέσματα της αναζήτησης εμφανίζονται στο Σύστημα Επισκόπησης του Μοντέλου επίσης.
- Συνοπτική Εικόνα Διαγράμματος (Overview Outline View): Είναι μία περιοχή επισκόπησης (view) στην οποίο εμφανίζεται σε κλίμακα το διάγραμμα το οποίο επεξεργάζεται ο χρήστης. Δημιουργεί δηλαδή μία πιο εποπτική αναπαράσταση του διαγράμματος. Σε μεγάλης έκτασης διαγράμματα δίνει την δυνατότητα της περιήγησης (scroll) στην επιφάνεια του διαγράμματος μέσω ενός «thumbnail».
- Πίνακας Ιδιοτήτων (Property Sheet): Είναι η υλοποίηση της υπό-μονάδας του Πίνακα Ιδιοτήτων (Property Sheet) όπως αυτή αναλύθηκε στο κεφάλαιο έξι της αρχιτεκτονικής του GRAMOFONE.
- Μπάρα Μηνυμάτων (Status Bar): Είναι η μπάρα του GRAMOFONE στην οποία εμφανίζονται μηνύματα λάθους κατά την διάρκεια της γραφικής επεξεργασίας ενός διαγράμματος.

Στη συνέχεια θα παρουσιαστούν και θα αναλυθούν σε βάθος τα επιμέρους τμήματα του Perspective που ορίζεται στο GRAMOFONE.

**Λειτουργικότητα του Μενού (Menu Bar):**

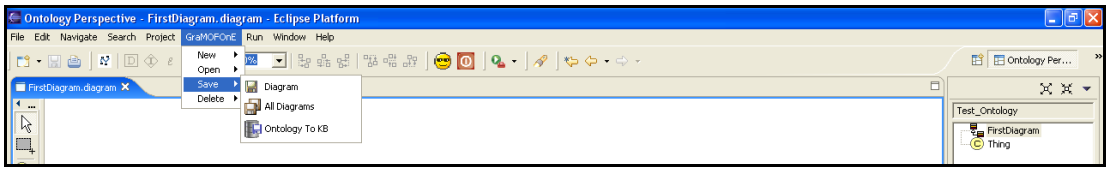
Στην ενότητα θα παρουσιαστούν οι δυνατές επιλογές και λειτουργίες που παρέχονται από το GRAMOFONE μέσω του μενού του γραφικού περιβάλλοντος.

Επιλέγοντας File→ GRAMOFONE ο χρήστης ενεργοποιεί το Perspective που ορίζεται στο GRAMOFONE και ουσιαστικά το ίδιο το GRAMOFONE. Η επιλογή του menu φαίνεται στην εικόνα 28.




**Εικόνα 28: Ενεργοποίηση Εργαλείου GRAMOFONE**

Στην εικόνα 29 παρουσιάζεται ένα στιγμιότυπο των επιλογών αποθήκευσης που παρέχει το μενού του GRAMOFONE.



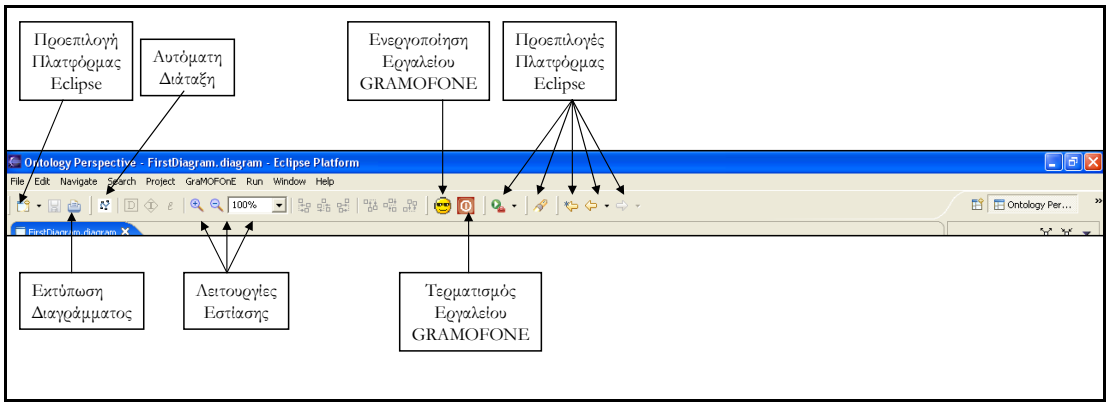

**Εικόνα 29: Μενού του GRAMOFONE**

Με τον ίδιο τρόπο ορίζονται και οι υπόλοιπες επιλογές του μενού. Οι διαθέσιμες επιλογές που παρέχονται από το μενού του GRAMOFONE είναι οι παρακάτω:

- Δημιουργία νέας οντολογίας. Η δημιουργία γίνεται σε αυτό το στάδιο στο τοπικό σύστημα αρχείων.
- Δημιουργία νέου διαγράμματος μέσα σε μία οντολογία.
- Ανάκτηση οντολογίας από τη Βάση Γνώσης του DBE.
- Ανάκτηση οντολογίας από το τοπικό σύστημα αρχείων.
- Αποθήκευση διαγράμματος στο τοπικό σύστημα αρχείων.
- Αποθήκευση όλων των ανακτημένων στην επιφάνεια εργασίας διαγραμμάτων στο τοπικό σύστημα αρχείων.
- Αποθήκευση οντολογίας στη Βάση Γνώσης του DBE.
- Διαγραφή οντολογίας από το τοπικό σύστημα αρχείων.
- Διαγραφή οντολογίας από τη Βάση Γνώσης του DBE.

**Λειτουργικότητα της Μπάρας Εργαλείων (Tool Bar):**

Στην ενότητα θα παρουσιαστούν οι δυνατές επιλογές και λειτουργίες που παρέχονται από το GRAMOFONE στη μπάρα εργαλείων του γραφικού περιβάλλοντος. Στην εικόνα 30 παρουσιάζεται ένα στιγμιότυπο της μπάρας εργαλείων.




**Εικόνα 30: Μπάρα Εργαλείων GRAMOFONE**

Η μπάρα εργαλείων βρίσκεται σε άμεση επικοινωνία και συνεργασία με τα διαγράμματα. Η μπάρα εργαλείων ενημερώνεται ανάλογα με τις επιλογές του χρήστη στο διάγραμμα. Η εικόνα 30 αποτελεί στιγμιότυπο της μπάρας εργαλείων όταν δεν έχει επιλεγεί κάποιο στοιχείο του διαγράμματος. Τα κουμπιά που είναι ενεργά παραμένουν ενεργά ανεξάρτητα από τις επιλογές του χρήστη στο διάγραμμα. Τα υπόλοιπα κουμπιά ενεργοποιούνται και απενεργοποιούνται ανάλογα με τις επιλογές του χρήστη πάνω στο διάγραμμα. Η λειτουργικότητα που παρέχεται με την χρήση των κουμπιών που είναι διαθέσιμα πάντα για τον χρήστη είναι η παρακάτω:

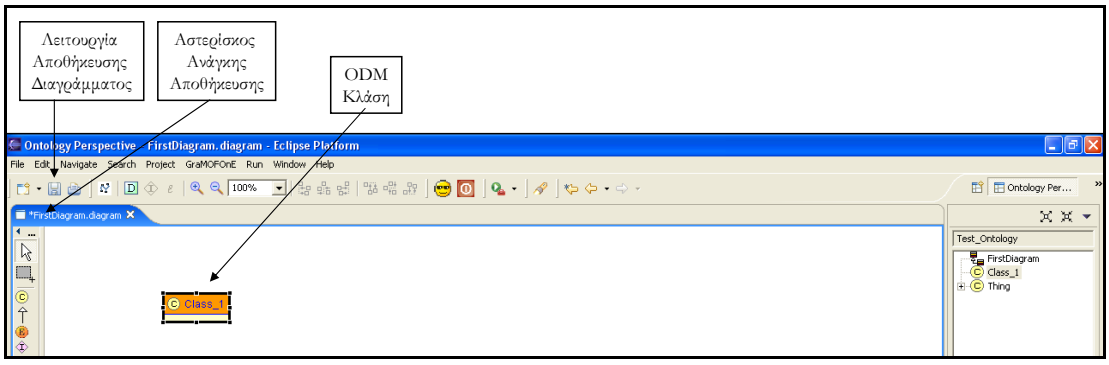
- Εκτύπωση Διαγράμματος: Είναι η λειτουργία εκτύπωσης ενός διαγράμματος. Ενεργοποιεί το προεπιλεγμένο (default) παράθυρο διαλόγου (dialog) του λειτουργικού συστήματος και εκτυπώνει το ενεργό διάγραμμα.
- Αυτόματη Διάταξη (Automatic Layout): Πρόκειται για μία λειτουργία η οποία δεν υλοποιήθηκε στο πλαίσιο της παρούσας διπλωματικής διατριβής αλλά χρησιμοποιήθηκε έτοιμη από το εργαλείο που είναι διαθέσιμο στο Διαδίκτυο στο πλαίσιο του άρθρου [29]. Ωστόσο πρόκειται για μία πολύ ενδιαφέρουσα και χρήσιμη λειτουργία, η οποία προσαρμόστηκε στις συνθήκες και τις ανάγκες του GRAMOFONE. Με τη χρήση της συγκεκριμένης λειτουργίας τα στοιχεία που περιέχονται μέσα στο ενεργό διάγραμμα τοποθετούνται με τρόπο βέλτιστο (αυτόματα). Ο χρήστης δεν έχει την δυνατότητα να επέμβει στην διάταξη (τακτοποίηση) των γραφικών στοιχείων του διαγράμματος καθώς αυτά τακτοποιούνται αυτόματα από το GRAMOFONE. Με την απενεργοποίηση της λειτουργίας ο χρήστης είναι πάλι σε θέση να επέμβει στην τοποθέτηση των γραφικών

στοιχείων του διαγράμματος. Η χρήση της λειτουργίας είναι προαιρετική. Η περαιτέρω ανάλυσή της δεν εντάσσεται στο πλαίσιο της παρούσας διπλωματικής διατριβής.

- Λειτουργίες Εστίασης:
    - Μεγέθυνση Διαγράμματος (Zoom In): Πρόκειται για την λειτουργία με την οποία ο χρήστης μπορεί να μεγεθύνει το ενεργό διάγραμμα.
    - Σμίκρυνση Διαγράμματος (Zoom Out): Πρόκειται για την λειτουργία με την οποία ο χρήστης μπορεί να σμικρύνει το ενεργό διάγραμμα.
    - Εστίαση σε συγκεκριμένο Ποσοστό ή Επίπεδο (Zoom Combo Box): Δίνεται η δυνατότητα στο χρήστη να εστιάσει στο ενεργό διάγραμμα χρησιμοποιώντας κάποια από τις δυνατές επιλογές που παρέχει το GRAMOFONE. Οι δυνατές επιλογές εστίασης είναι 25% - 2000%, εστίαση με βάση το διάγραμμα, εστίαση με βάση το ύψος του διαγράμματος ή εστίαση με βάση το πλάτος του διαγράμματος .
- Ενεργοποίηση Εργαλείου GRAMOFONE: Ενεργοποίηση του Perspective που ορίζεται στο GRAMOFONE, δηλαδή ουσιαστικά ενεργοποίηση του ίδιου του εργαλείου GRAMOFONE.
- Τερματισμός Εργαλείου GRAMOFONE: Το GRAMOFONE κλείνει όλα τα ενεργά διαγράμματα του περιβάλλοντος εργασίας και τερματίζει τη λειτουργία της πλατφόρμας Eclipse.
- Οι προεπιλεγμένες (default) επιλογές παρέχονται από την πλατφόρμα Eclipse και δεν παρέχουν λειτουργικότητα χρήσιμη για το GRAMOFONE.
- Κάθε κουμπί συνοδεύεται από ένα επεξηγηματικό μήνυμα (tool tip) που παρέχει μία σύντομη περιγραφή της λειτουργίας.

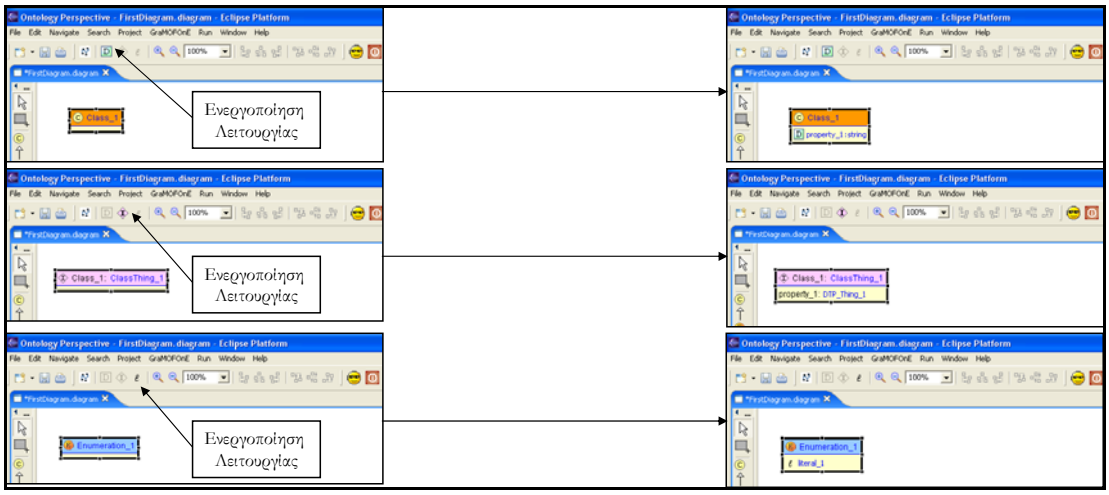
Τα υπόλοιπα κουμπιά των οποίων η ενεργοποίηση εξαρτάται από τις επιλογές του χρήστη πάνω στο διάγραμμα θα παρουσιαστούν στην συνέχεια.

Όταν κάποιο διάγραμμα τροποποιηθεί και το περιεχόμενό του δεν έχει αποθηκευτεί ώστε να περιλαμβάνει τις τροποποιήσεις τότε ενεργοποιείται το κουμπί της αποθήκευσης του διαγράμματος στο τοπικό σύστημα αρχείων που προτρέπει τον χρήστη να αποθηκεύσει το διάγραμμα. Παράλληλα εμφανίζεται και ένας αστερίσκος στο πάνω αριστερό τμήμα του διαγράμματος δίπλα στο όνομα του διαγράμματος που εκφράζει το γεγονός πως το διάγραμμα χρειάζεται αποθήκευση. Το κουμπί της αποθήκευσης φαίνεται στην εικόνα 31. Στην εικόνα 31 προστέθηκε μία ODM Κλάση σε ένα διάγραμμα χωρίς αυτό να αποθηκευτεί.



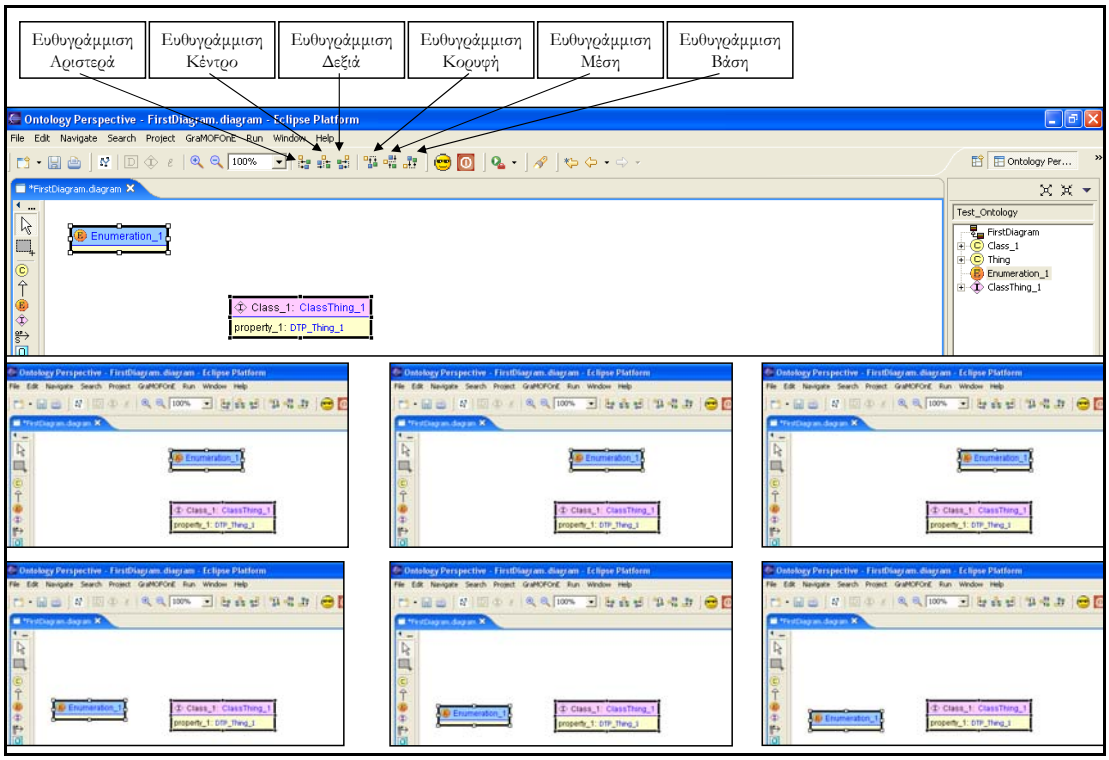

**Εικόνα 31: Λειτουργία Αποθήκευσης Διαγράμματος**

Στο κεφάλαιο περιγραφής του μετά-μοντέλου ODM (κεφάλαιο 4) σημειώθηκε πως για μία ODM Κλάση μπορούν να οριστούν μία ή περισσότερες Ιδιότητες τύπου «Datatype Property», για ένα Στιγμιότυπο μίας Κλάσης (ClassThing) μπορούν να οριστούν ένα ή περισσότερα Στιγμιότυπα Ιδιότητας τύπου «Datatype Property» (DatatypePropertyThing) και πως για μία Απαρίθμηση (Enumeration) μπορούν να οριστούν ένα ή περισσότερα «Plain Literals». Για την γραφική διεκπεραίωση αυτών των λειτουργιών υπάρχουν τρία κουμπιά στη μπάρα εργαλείων που αναλαμβάνουν αντίστοιχα τις ενέργειες που προαναφέρθηκαν. Για παράδειγμα όταν έχει επιλεγεί στο διάγραμμα μία ODM Κλάση, ενεργοποιείται το κουμπί που δημιουργεί μία Ιδιότητα τύπου «Datatype Property» η οποία έχει σαν «domain» την ODM Κλάση. Ανάλογα λειτουργούν και τα άλλα δύο κουμπιά. Ότι συζητήθηκε μέχρι εδώ φαίνεται στην εικόνα 32. Στην εικόνα φαίνεται και πώς τροποποιούνται τα διαγράμματα μετά τη χρήση αυτών των κουμπιών.




**Εικόνα 32: Λειτουργίες για Κλάσεις, Στιγμιότυπα Κλάσεων και Απαριθμήσεων**

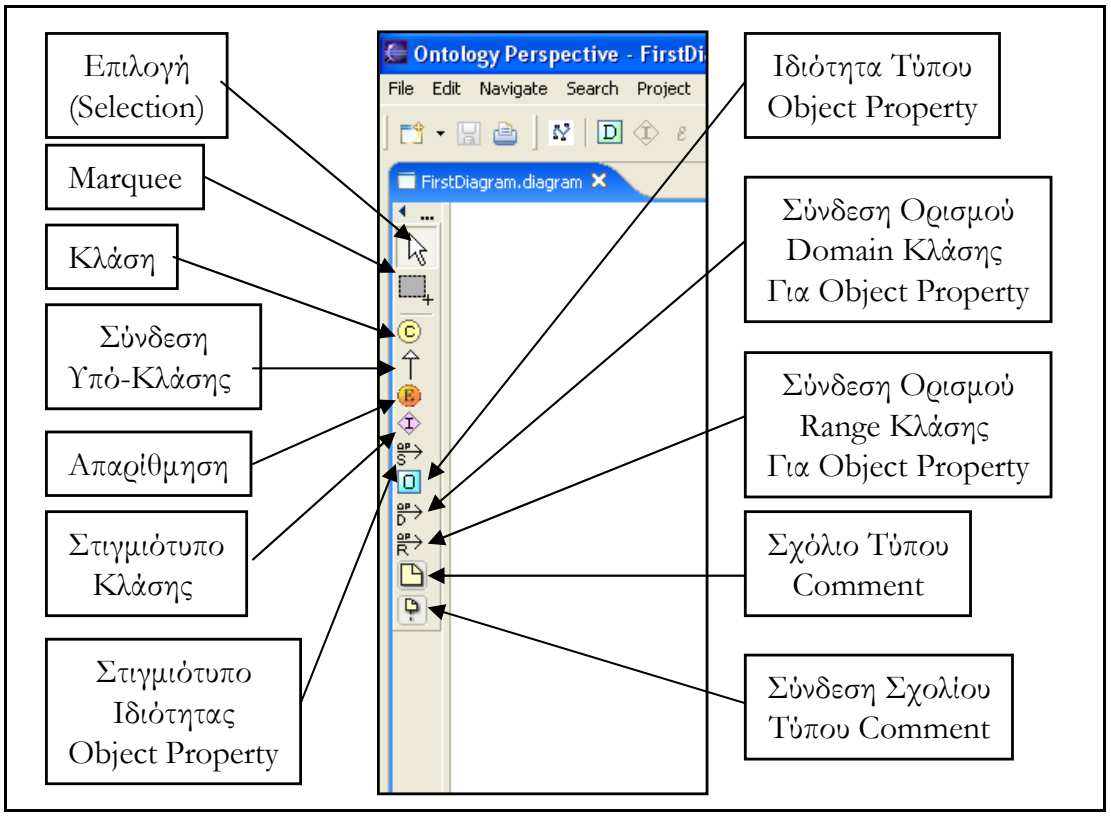
Κάθε φορά που επιλέγονται δύο ή περισσότερα γραφικά στοιχεία σε κάποιο διάγραμμα ενεργοποιούνται τα κουμπιά ευθυγράμμισης. Τα κουμπιά αυτά καθώς και το γραφικό αποτέλεσμα της χρήσης τους φαίνεται στην εικόνα 33.




**Εικόνα 33: Λειτουργίες Ευθυγράμμισης**

### Περιγραφή Παλέτας Γραφικού Επεξεργαστή Διαγράμματος (GEF Editor)

Στην ενότητα θα περιγραφεί η παλέτα που εμφανίζεται σε κάθε διάγραμμα και από την οποία μπορούν να προστεθούν στοιχεία μέσα στο διάγραμμα. Η παλέτα και οι επιλογές που παρέχονται σε αυτήν φαίνονται στην εικόνα 34. Θεωρούνται πλέον γνωστές οι έννοιες που ορίζονται στο μετά-μοντέλο ODM που αναλύθηκαν στο τέταρτο κεφάλαιο.



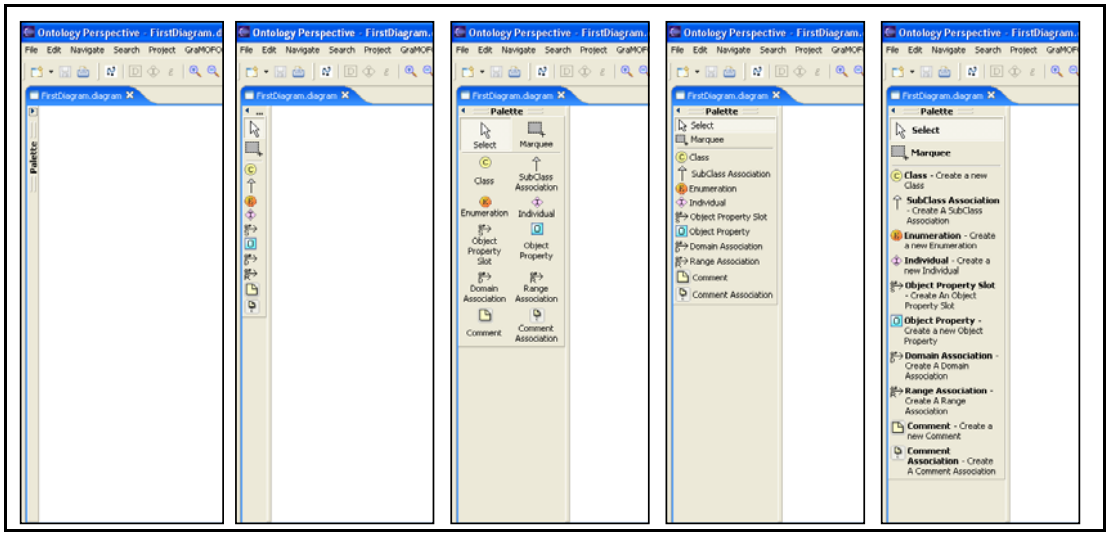

**Εικόνα 34: Παλέτα Γραφικού Επεξεργαστή GRAMOFONE**

Η παλέτα που εμφανίζεται στα διαγράμματα όπως φαίνεται και στην εικόνα 34 παρέχει τις εξής λειτουργίες:

- Δυνατότητα επιλογής (selection) γραφικών όρων πάνω στα διαγράμματα.
- Δυνατότητα επιλογής ενός τμήματος σε ένα διάγραμμα και των στοιχείων που περιέχονται στο τμήμα αυτό (marquee).
- Δυνατότητα δημιουργίας ODM Κλάσης.
- Δυνατότητα σύνδεσης δύο ODM Κλάσεων με την σύνδεση υπό-κλάσης για την δημιουργία ιεραρχών κλάσεων.
- Δημιουργία Απαρίθμησης (Enumeration).
- Δημιουργία Στιγμιότυπου Κλάσης (ClassThing).
- Δημιουργία Στιγμιότυπου Ιδιότητας τύπου «Object Property» (ObjectPropertyThing).
- Δημιουργία Ιδιότητας τύπου «Object Property».
- Σύνδεση μίας ODM Κλάσης με μία Ιδιότητα τύπου «Object Property» με σκοπό τον ορισμό των domain κλάσεων της Ιδιότητας αυτής.
- Σύνδεση μία Ιδιότητας τύπου «Object Property» με μία ODM Κλάση με σκοπό τον ορισμό των range κλάσεων της Ιδιότητας αυτής.

- Δημιουργία Σχολίου (Annotation) τύπου «Comment». Μόνο σχόλια τύπου «Comment» μπορούν να οριστούν γραφικά.
- Σύνδεση ενός Σχολίου (Annotation) τύπου «Comment» με το στοιχείο στο οποίο απευθύνεται μέσω αυτής της σύνδεσης.

Η παλέτα επίσης έχει την ιδιότητα να προσαρμόζει την εμφάνισή της ανάλογα με την επιλογή του χρήστη. Έχει τέσσερις δυνατές μορφές καθώς επίσης και την δυνατότητα να «κρύβεται» και να «εμφανίζεται» όποτε ο χρήστης το επιθυμεί. Οι δυνατές μορφές της παλέτας καθώς και η ιδιότητά της να «κρύβεται» φαίνονται στην εικόνα 35. Στην εικόνα φαίνονται με τη σειρά οι εξής μορφές της παλέτας: «κρυμμένη», «μόνο εικονίδια», «εικονίδια και τίτλοι εικονιδίων», «λίστα επιλογών» και «λίστα επιλογών με λεπτομέρειες».

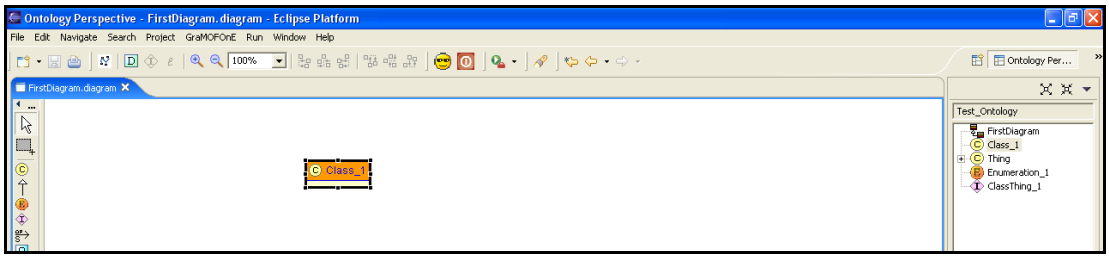


**Εικόνα 35: Μορφές Παλέτας**

### Γραφική Αναπαράσταση Εννοιών του Μοντέλου (Οντολογίας)

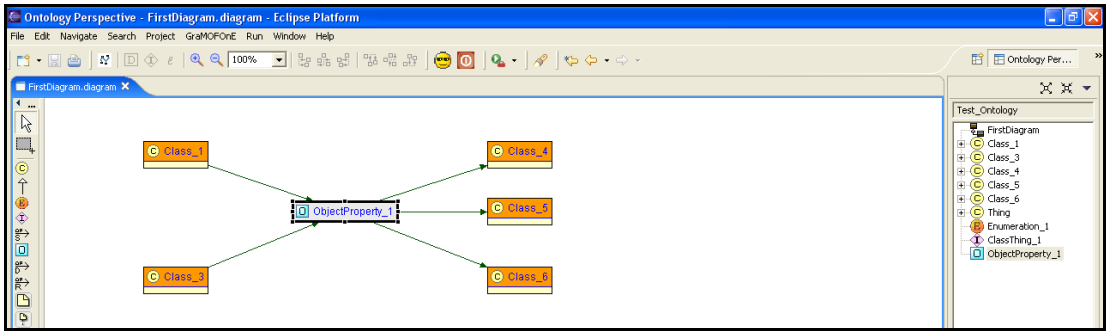
Στην ενότητα θα παρουσιαστούν οι γραφικές αναπαραστάσεις των εννοιών που ορίζονται στο μετά-μοντέλο ODM και υλοποιήθηκαν στο GRAMOFONE.

Η έννοια της οντολογίας δεν έχει γραφική αναπαράσταση καθώς αποτελεί σημασιολογικά ένα πλαίσιο (container / namespace) το οποίο περιλαμβάνει τους όρους που ορίζονται στην οντολογία αυτή. Η γραφική αναπαράσταση της έννοιας της ODM Κλάσης φαίνεται στην εικόνα 36.



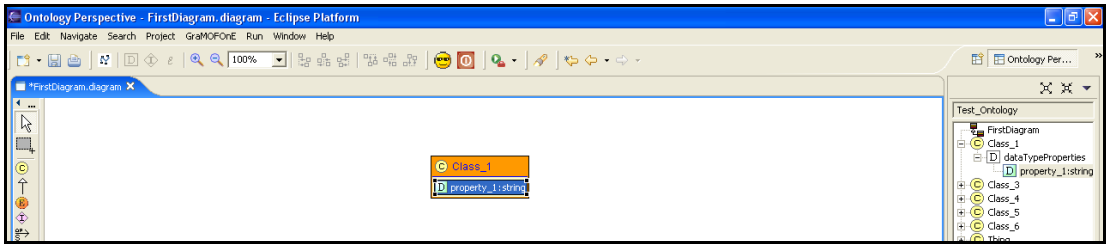

**Εικόνα 36: Γραφική αναπαράσταση ODM Κλάσης**

Η έννοια της Ιδιότητας τύπου «Object Property» φαίνεται στην εικόνα 37. Στην εικόνα φαίνεται μία Ιδιότητα τύπου «Object Property» η οποία έχει συνδεθεί με δύο domain ODM Κλάσεις και τρεις range ODM Κλάσεις.




**Εικόνα 37: Γραφική αναπαράσταση Ιδιότητας τύπου «Object Property»**

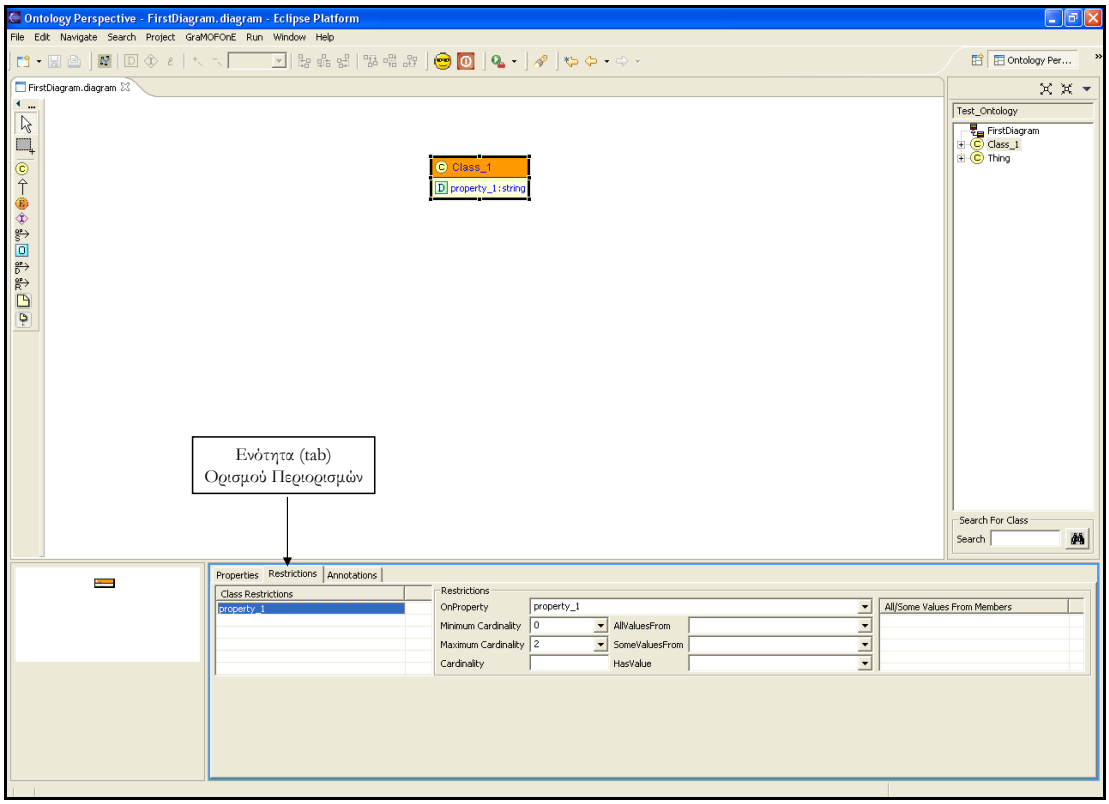
Η έννοια της Ιδιότητας τύπου «Datatype Property» φαίνεται στην εικόνα 38. Παρατηρούμε πως οι Ιδιότητες τύπου «Datatype Property» αναπαριστώνται γραφικά σαν «παιδιά» των ODM Κλάσεων (domain κλάσεις των Ιδιοτήτων). Για παράδειγμα στην εικόνα 38 φαίνεται μία Ιδιότητα τύπου «Datatype Property» με όνομα (name) «property_1» και εύρος τιμών (DataRange) «string». Επιπλέον η Ιδιότητα έχει σαν domain την ODM Κλάση με όνομα (name) Class_1.




**Εικόνα 38: Γραφική αναπαράσταση Ιδιότητας τύπου «Datatype Property»**

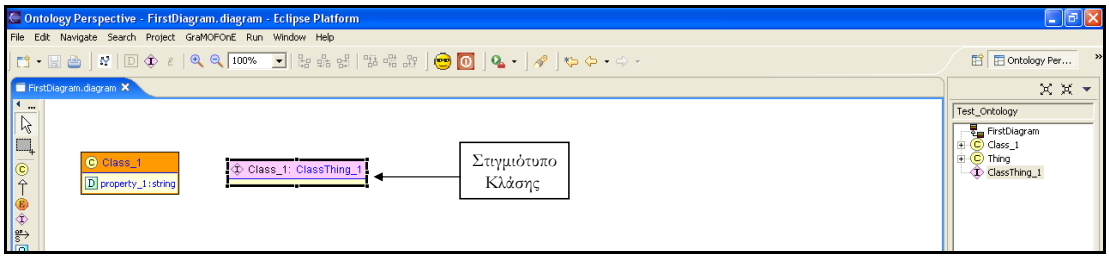
Η έννοια του Περιορισμού (Restriction) ορίζεται μέσω του Πίνακα Ιδιοτήτων (Property Sheet). Κάθε Περιορισμός ορίζεται για μία συγκεκριμένη κλάση. Για τον λόγο αυτό στον Πίνακα Ιδιοτήτων των κλάσεων υπάρχει μία ενότητα (tab) με τίτλο «Restrictions». Μέσα σε αυτή την ενότητα ο χρήστης μπορεί να ορίσει τους Περιορισμούς που ισχύουν για την συγκεκριμένη κλάση. Για παράδειγμα στην εικόνα 39 ορίστηκε για την κλάση του διαγράμματος ένας περιορισμός πάνω (onProperty) στην Ιδιότητα τύπου «Datatype Property» με όνομα «property_1» η οποία όπως φαίνεται στην εικόνα έχει domain την

κλάση του διαγράμματος. Ο περιορισμός ορίζει πως κάθε Στιγμιότυπο της κλάσης (ClassThing) μπορεί να έχει το πολύ μέχρι δύο Στιγμιότυπα της Ιδιότητας στην οποία αναφέρεται ο περιορισμός (DatatypePropertyThing).




**Εικόνα 39: Γραφική αναπαράσταση Περιορισμών**

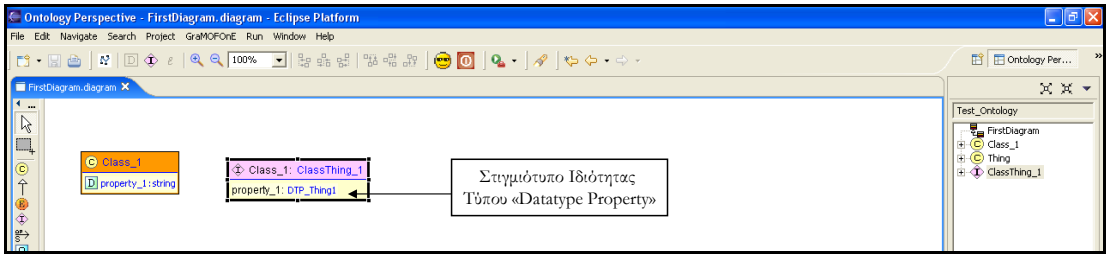
Η έννοια του Στιγμιότυπου Κλάσης (ClassThing) φαίνεται στην εικόνα 40. Στην εικόνα έχει οριστεί ένα Στιγμιότυπο για την κλάση με όνομα «Class_1». Το Στιγμιότυπο έχει όνομα «ClassThing_1». Ο τίτλος της γραφικής αναπαράστασης του Στιγμιότυπου Κλάσης είναι της μορφής «Όνομα Κλάσης Στιγμιότυπου (HasType Class): Όνομα Στιγμιότυπου».




**Εικόνα 40: Γραφική αναπαράσταση Στιγμιότυπου Κλάσης**

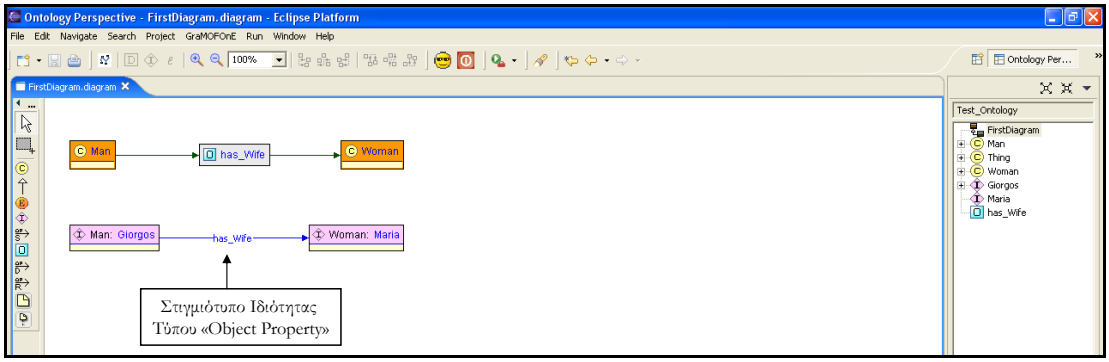
Η έννοια του Στιγμιότυπου Ιδιότητας τύπου «Datatype Property» (DatatypePropertyThing) φαίνεται στην εικόνα 41. Παρατηρούμε πως τα Στιγμιότυπα αυτού του είδους αναπαριστώνται γραφικά σαν «παιδιά» των Στιγμιότυπων Κλάσης (τα οποία αποτελούν domains). Για παράδειγμα στην εικόνα 41 φαίνεται ένα Στιγμιότυπο Ιδιότητας τύπου «Datatype Property» (DatatypePropertyThing) με τιμή (range) «DTP_Thing1». Το Στιγμιότυπο έχει σαν τύπο (δηλαδή δημιουργήθηκε σαν στιγμιότυπο

της Ιδιότητας) την Ιδιότητα (Datatype Property) με όνομα «property_1». Ο τίτλος της γραφικής αναπαράστασης των Στιγμιότυπων αυτού του είδους (DatatypePropertyThing) είναι της μορφής «Όνομα Ιδιότητας Στιγμιότυπου (HasType Datatype Property): Τιμή (range) Στιγμιότυπου».



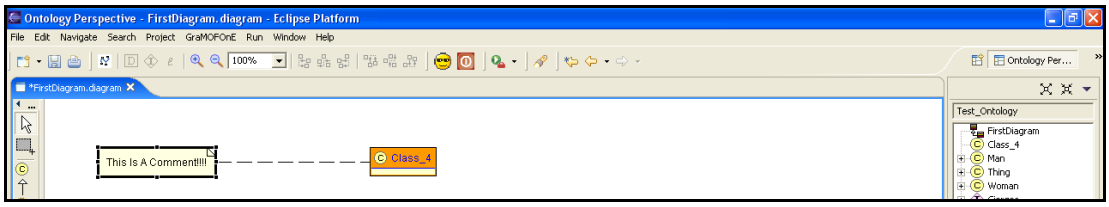

**Εικόνα 41: Γραφική αναπαράσταση Στιγμιότυπου Ιδιότητας τύπου «Datatype Property»**

Η έννοια του Στιγμιότυπου Ιδιότητας τύπου «Object Property» (ObjectPropertyThing) φαίνεται στην εικόνα 42. Κάθε Στιγμιότυπο αυτού του είδους πρέπει υποχρεωτικά να ορίζει την Ιδιότητα (Object Property) που είναι ο τύπος του, δηλαδή της οποίας είναι στιγμιότυπο. Τα Στιγμιότυπα αυτού του είδους υλοποιήθηκαν σαν συνδέσεις οι οποίες συνδέουν τα Στιγμιότυπα Κλάσεων στα οποία αναφέρονται. Για παράδειγμα στην εικόνα 42 έχουν οριστεί δύο κλάσεις με όνομα «Man» και «Woman», οι οποίες συνδέονται με μία Ιδιότητα (Object Property) με όνομα «has_Wife» με αντίστοιχη σημασιολογία. Στην συνέχεια ορίζονται δύο Στιγμιότυπα των κλάσεων με όνομα «Giorgos» και «Maria» αντίστοιχα. Για να δηλωθεί πως το στιγμιότυπο «Giorgos» έχει γυναίκα το στιγμιότυπο «Maria», ορίζεται ένα Στιγμιότυπο της Ιδιότητας με την σημασιολογία αυτή. Το στιγμιότυπο έχει την μορφή σύνδεσης και φαίνεται στην εικόνα 42.



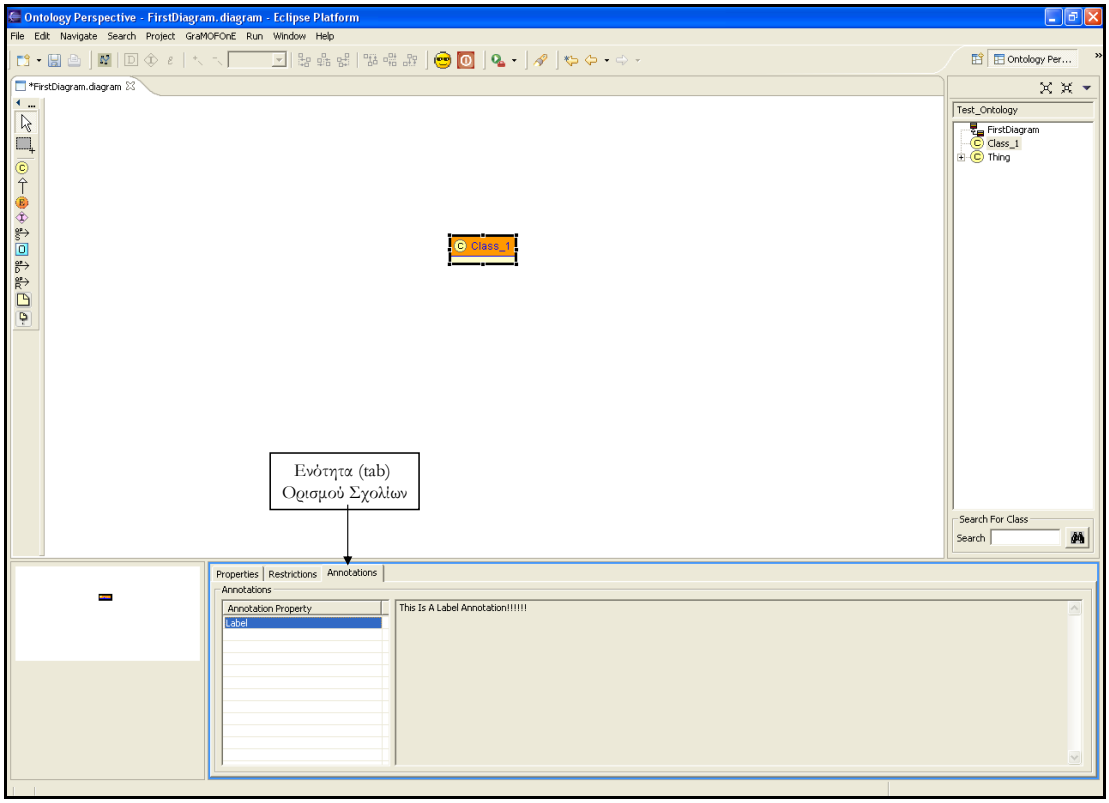

**Εικόνα 42: Γραφική αναπαράσταση Στιγμιότυπου Ιδιότητας τύπου «Object Property»**

Η γραφική αναπαράσταση της έννοιας του Σχολίου (Annotation) φαίνεται στην εικόνα 43. Με τον τρόπο αυτό μπορούν να οριστούν μόνο Σχόλια του τύπου «Comment». Κάθε Σχόλιο συνδέεται γραφικά με την έννοια που σχολιάζει μέσω της σύνδεσης «Comment Association» της παλέτας. Στην εικόνα 43 φαίνεται ένα Σχόλιο το οποίο απευθύνεται σε μία ODM Κλάση με όνομα “Class_4”.



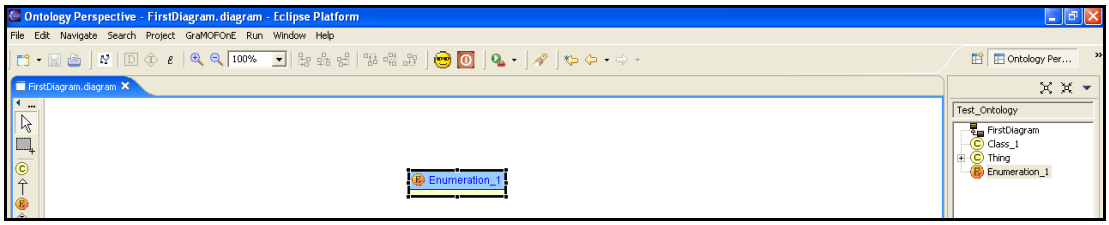

**Εικόνα 43: Γραφική αναπαράσταση Σχολίων τύπου «Comment»**

Οι υπόλοιποι τύποι σχολίων ορίζονται μέσω του Πίνακα Ιδιοτήτων. Κάθε έννοια της οντολογίας που μπορεί να σχολιαστεί έχει μία ενότητα (tab) στον Πίνακα Ιδιοτήτων της με τίτλο «Annotations» στην οποία ο χρήστης μπορεί να προσθέσει σχόλια για την έννοια αυτή. Για παράδειγμα στην εικόνα 44 φαίνεται μία κλάση με όνομα «Class_1» η οποία περιέχει ένα Σχόλιο τύπου «Label» με κείμενο «This Is A Label Annotation!!!!!!».



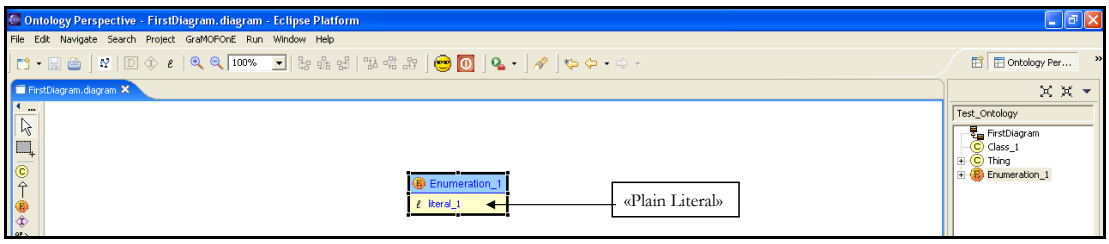

**Εικόνα 44: Γραφική αναπαράσταση Σχολίων**

Στην εικόνα 45 φαίνεται η γραφική αναπαράσταση της έννοιας της Απαρίθμησης (Enumeration).



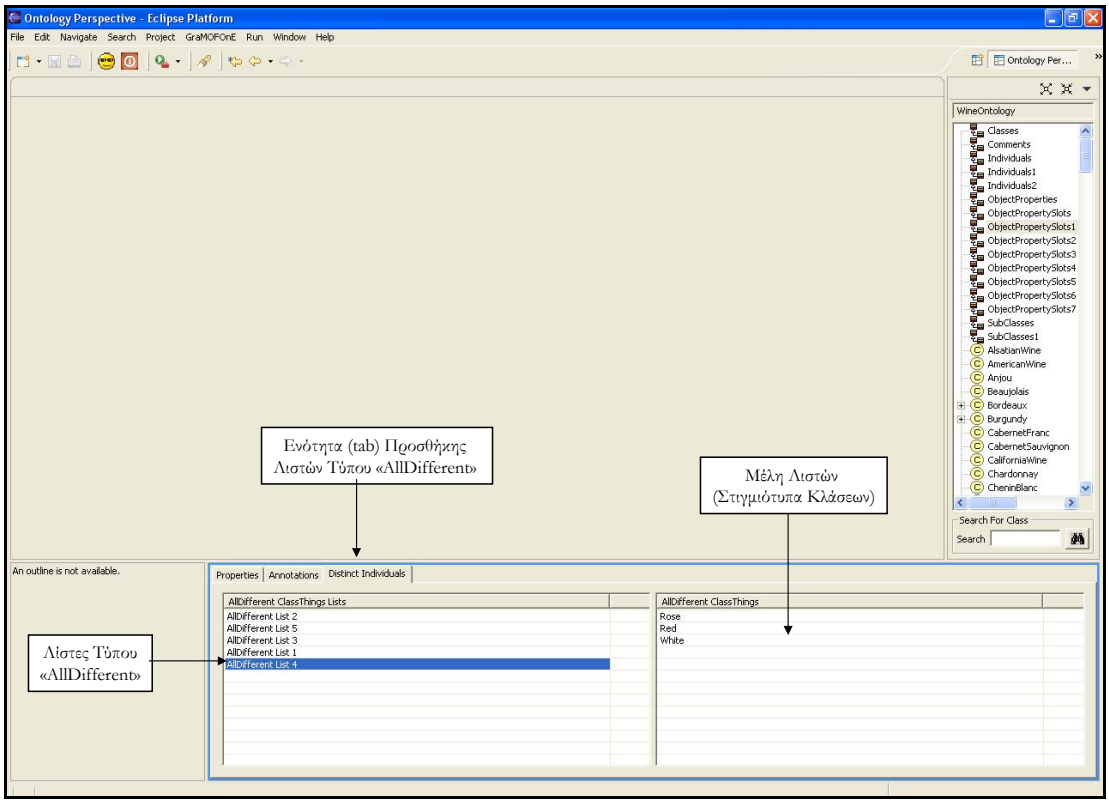

**Εικόνα 45: Γραφική αναπαράσταση Απαριθμήσεων**

Η γραφική αναπαράσταση της έννοιας «Plain Literal» φαίνεται στην εικόνα 46. Παρατηρούμε πως τα «Plain Literals» αναπαριστώνται γραφικά σαν παιδιά των Απαριθμήσεων (Enumerations) τα οποία τα περιέχουν. Για παράδειγμα στην εικόνα 46 φαίνεται ένα «Plain Literal» με τιμή (lexicalForm) «literal_1», το οποίο περιέχεται σε μία Απαρίθμηση με όνομα (name) "Enumeration_1".




**Εικόνα 46: Γραφική αναπαράσταση «Plain Literal»**

Ο ορισμός της έννοιας των λιστών «AllDifferent», δηλαδή των λιστών των οποίων τα μέλη (Στιγμιότυπα Κλάσεων - ClassThings) ορίζονται πως δεν έχουν κοινά σημεία, πραγματοποιείται μέσω του Πίνακα Ιδιοτήτων της οντολογίας, μέσω της ενότητας (tab) με τίτλο «Distinct Individuals». Για παράδειγμα στην εικόνα 47, η οποία αποτελεί στιγμιότυπο (screenshot) της πρότυπης οντολογίας (οντολογία οινολογίας) που θα παρουσιαστεί στο επόμενο κεφάλαιο, φαίνεται μία λίστα τύπου «AllDifferent» με μέλη τρία Στιγμιότυπα Κλάσεων «Rose», «Red» και «White» τα οποία είναι στιγμιότυπα της κλάσης «WineColor» και τα οποία προφανώς είναι διαφορετικά μεταξύ τους.

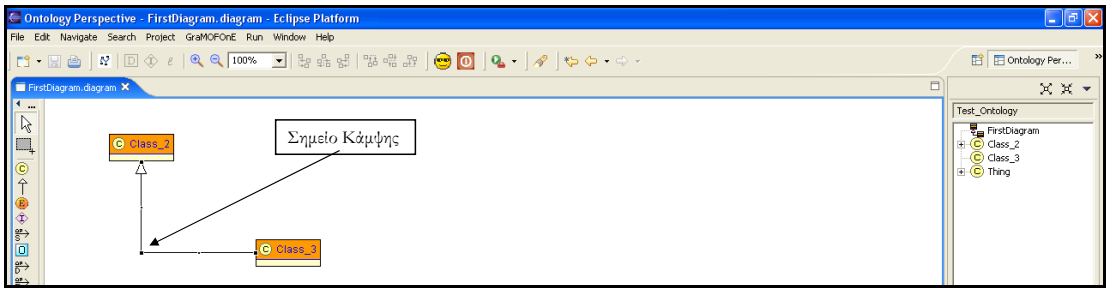



**Εικόνα 47: Γραφική αναπαράσταση Λιστών "AllDifferent"**

### Επιπρόσθετες Ιδιότητες Διαγραμμάτων

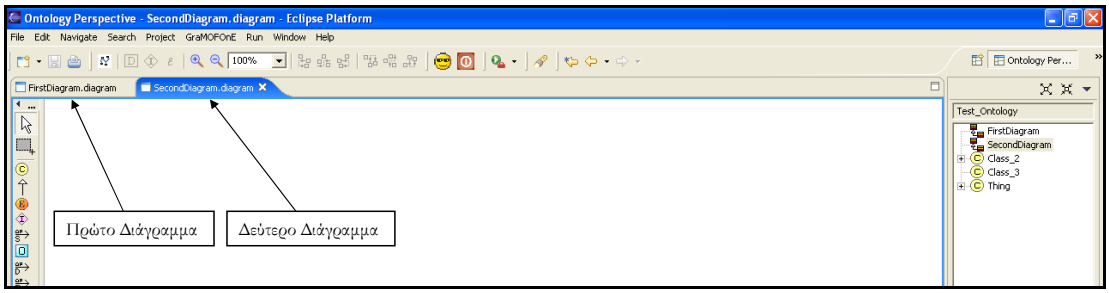
Κάθε είδους σύνδεση (connection) που δημιουργείται σε κάποιο διάγραμμα έχει επιπλέον δύο ιδιότητες:

- Κάθε σύνδεση έχει «σημεία κάμψης» (BendPoints). Τα σημεία κάμψης είναι σημεία πάνω στην γραμμή που ορίζεται από την σύνδεση τα οποία μπορούν εκ νέου να μετατοπιστούν δημιουργώντας με αυτό τον τρόπο τεθλασμένες συνδέσεις. Παράδειγμα τέτοιας σύνδεσης φαίνεται στην εικόνα 48.
- Κάθε σύνδεση έχει την ιδιότητα να "γνωρίζει" τι είδους στοιχεία συνδέει, επιτρέποντας μόνο τις συνδέσεις μεταξύ των επιτρεπόμενων εννοιών και αποτρέποντας το χρήστη από την λανθασμένη χρήση των συνδέσεων. Για παράδειγμα η σύνδεση ορισμού υπό-κλάσεων συνδέει μόνο ODM Κλάσεις. Αν ο χρήστης προσπαθήσει να συνδέσει μία ODM Κλάση με μία Ιδιότητα τύπου «Object Property», το GRAMOFONE δεν θα επιτρέψει την σύνδεση αυτή εμφανίζοντας ένα φιλικό προς τον χρήστη εικονίδιο που τον ενημερώνει πως δεν επιτρέπεται τέτοιου είδους σύνδεση.




**Εικόνα 48: Παράδειγμα Σημείου Κάμψης**

Επιπλέον στο GRAMOFONE είναι δυνατή η ανάκτηση παραπάνω του ενός διαγραμμάτων στο περιβάλλον εργασίας. Μία τέτοια περίπτωση φαίνεται στην εικόνα 49 στην οποία είναι ανακτημένα δύο διαγράμματα.



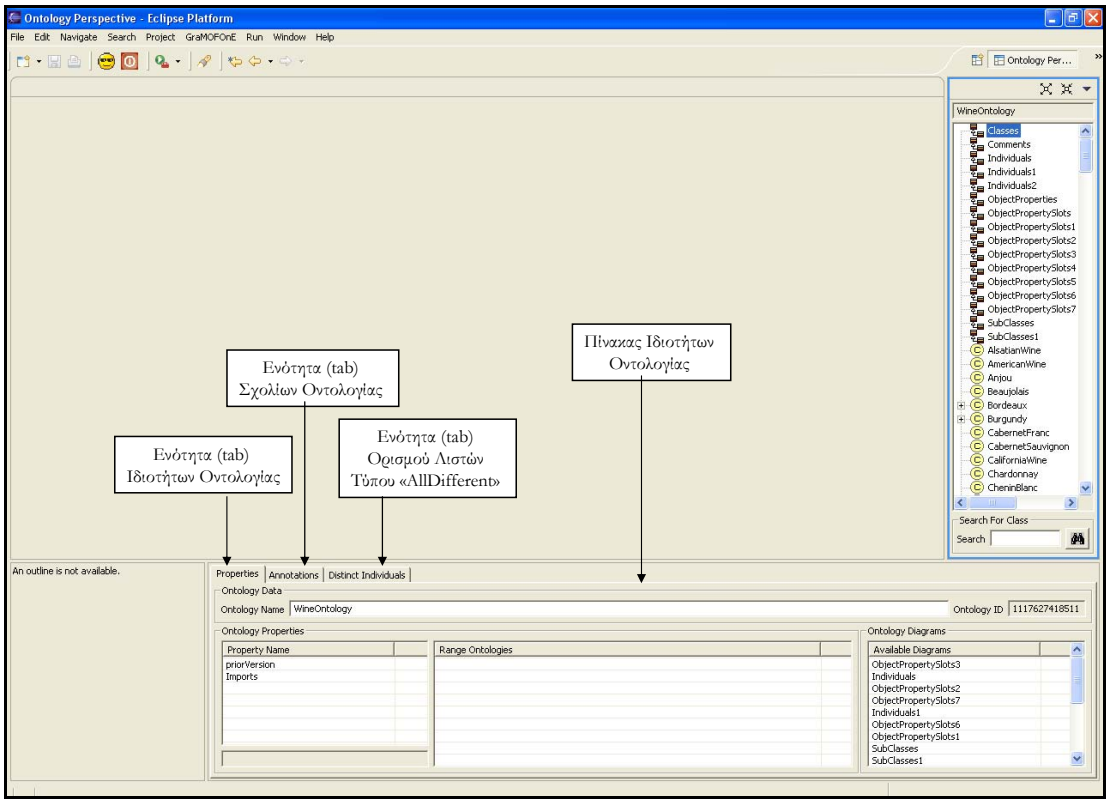

**Εικόνα 49: Πολλαπλά Διαγράμματα**

### Αναπαράσταση των Ιδιοτήτων των Εννοιών του ODM στο GRAMOFONE

Οι πληροφορίες που παίρνει ο χρήστης από τα διαγράμματα (από την γραφική αναπαράσταση των εννοιών) για το κάθε στοιχείο που δημιουργεί είναι ελάχιστες και

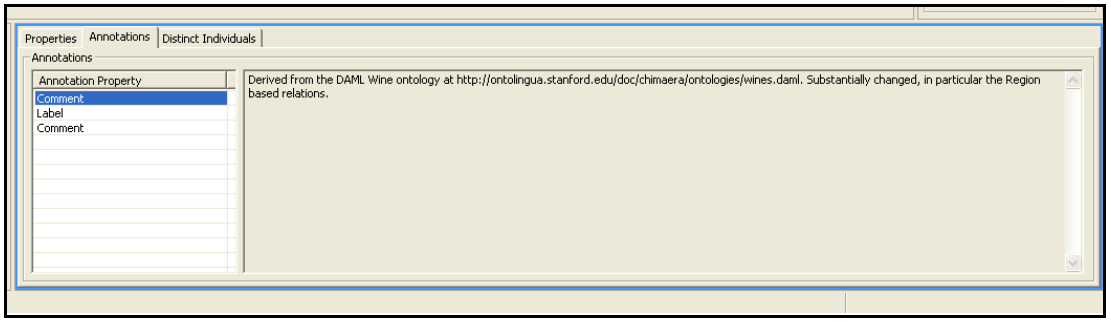
σίγουρα δεν καλύπτουν τις ιδιότητες που ορίζει το ODM για κάθε έννοια. Για την πλήρη κάλυψη των ιδιοτήτων των εννοιών που ορίζονται από το ODM δημιουργήθηκε ο Πίνακας Ιδιοτήτων (Property Sheet). Ο Πίνακας Ιδιοτήτων (κεφάλαιο 6) έχει την ικανότητα ανά πάσα στιγμή να δείχνει τις ιδιότητες που ορίζονται από το ODM για οποιοδήποτε στοιχείο το οποίο θα επιλεχτεί από το χρήστη, είτε σε κάποιο διάγραμμα είτε από στο Σύστημα Επισκόπησης του Μοντέλου (Model Explorer). Επιπλέον δείχνει όπως είναι λογικό και τις τιμές των ιδιοτήτων του συγκεκριμένου στοιχείου που επιλέχθηκε. Ο χρήστης ακόμα έχει την δυνατότητα να τροποποιήσει τις τιμές αυτές όπως εκείνος επιθυμεί. Στην συνέχεια θα παρουσιαστούν στιγμιότυπα (screenshots) με τα πιθανά περιεχόμενα του Πίνακα Ιδιοτήτων (Property Sheet) ανάλογα με το στοιχείο που έχει επιλεχθεί. Τα περιεχόμενα κάθε έννοιας του Πίνακα Ιδιοτήτων ανταποκρίνονται πλήρως στις ανάγκες αναπαράστασης του μοντέλου (οντολογία) το οποίο ορίζεται στο μετά-μοντέλο ODM. Τα στιγμιότυπα (screenshots) έχουν παρθεί από την υλοποιημένη, για τις ανάγκες της αξιολόγησης του GRAMOFONE, πρότυπης οντολογίας οινολογίας (Wine Ontology) η οποία θα παρουσιαστεί στο επόμενο κεφάλαιο.

Η έννοια της οντολογίας δεν έχει κάποια γραφική αναπαράσταση. Ωστόσο κάθε οντολογία χαρακτηρίζεται από κάποιες συγκεκριμένες ιδιότητες οι οποίες ορίζονται από το ODM. Για την κάλυψη αυτών των ιδιοτήτων αλλά και κάποιων επιπλέον ιδιοτήτων οι οποίες διευκολύνουν το χρήστη στη δημιουργία οντολογιών, προβλέφθηκε πως όποτε ο χρήστης επιλέξει (κάνει αριστερό κλικ με το ποντίκι) τον καμβά στον οποίο εισάγονται τα γραφικά στοιχεία ή κάποιο διάγραμμα στο Σύστημα Επισκόπησης του Μοντέλου (Model Explorer), τότε ο Πίνακας Ιδιοτήτων θα παρουσιάζει τις ιδιότητες τις οντολογίας. Τα περιεχόμενα του Πίνακα Ιδιοτήτων σε αυτή την περίπτωση φαίνονται στην εικόνα 50. Να σημειωθεί πως ο Πίνακας Ιδιοτήτων, επειδή ορισμένες φορές περιλαμβάνει πολλά στοιχεία τα οποία δεν χωράνε στην ορατή επιφάνεια του Πίνακα Ιδιοτήτων, διαθέτει μία μπάρα (scroll bar) με την οποία δίνεται η δυνατότητα στο χρήστη να μετακινεί την ορατή επιφάνεια του Πίνακα Ιδιοτήτων.

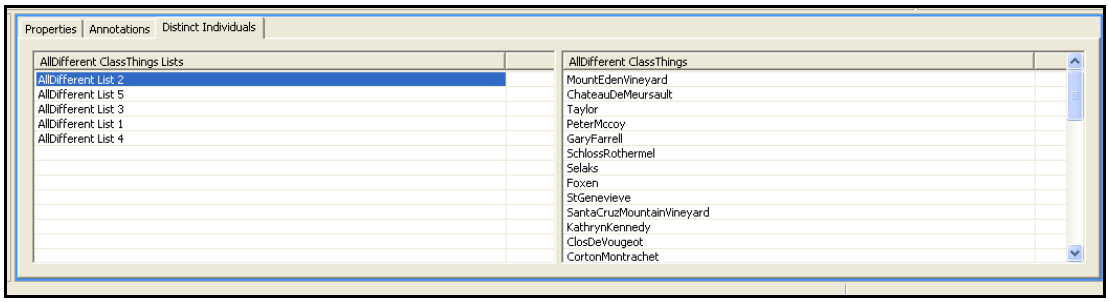



**Εικόνα 50: Πίνακας Ιδιοτήτων Οντολογίας**

Ο Πίνακας Ιδιοτήτων της Οντολογίας διαθέτει, όπως φαίνεται στην εικόνα 50, μία δεύτερη ενότητα (tab) με τίτλο «Annotations» στην οποία φαίνονται τα σχόλια που έχουν προστεθεί για την οντολογία, και μία ενότητα (tab) με τίτλο «Distinct Individuals» στην οποία φαίνονται οι λίστες τύπου «AllDifferent» που έχουν οριστεί στην οντολογία. Τα περιεχόμενα των δύο ενοτήτων στο πλαίσιο της πρότυπης οντολογίας οινολογίας (wine ontology) φαίνονται στις εικόνες 51 και 52 αντίστοιχα.

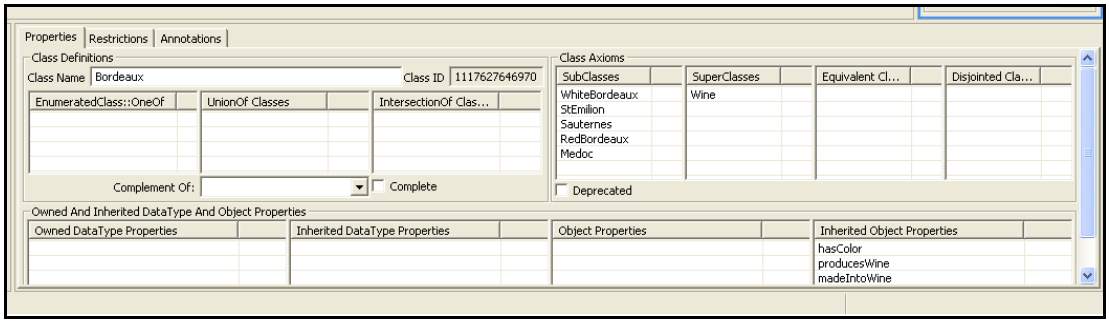



**Εικόνα 51: Ενότητα Σχολίων Πίνακα Ιδιοτήτων Οντολογίας**



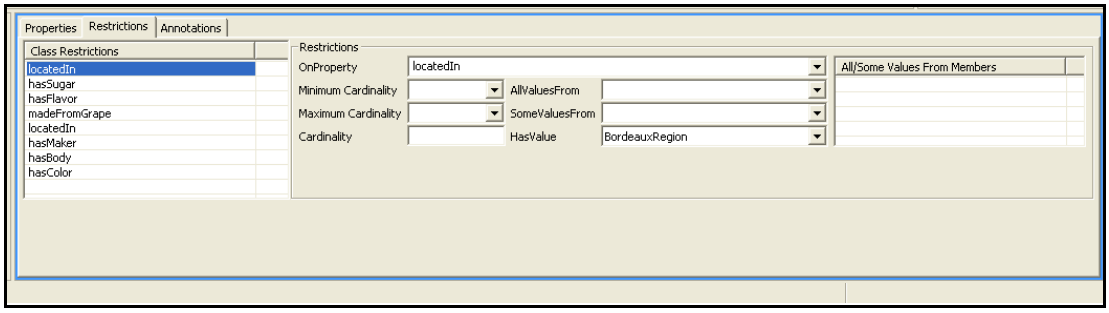

**Εικόνα 52: Ενότητα Λιστών "AllDifferent" Οντολογίας**

Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποια ODM Κλάση μίας οντολογίας φαίνονται στην εικόνα 53.



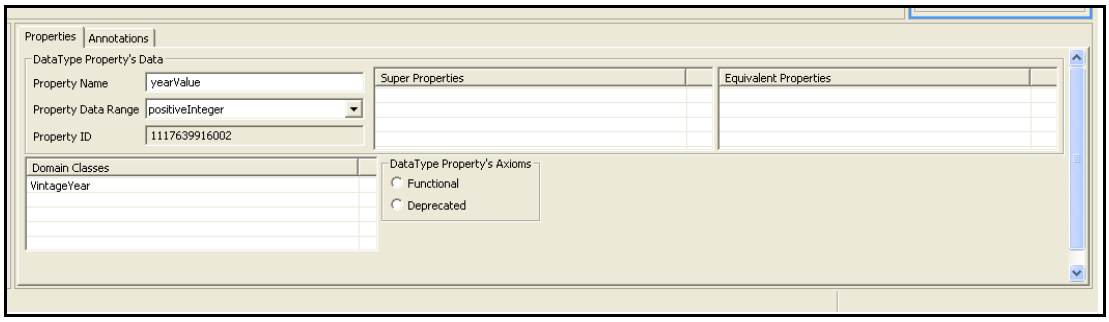

**Εικόνα 53: Πίνακας Ιδιοτήτων ODM Κλάσης**

Ο Πίνακας Ιδιοτήτων των ODM Κλάσεων έχει επιπλέον δύο ενότητες (tabs). Την ενότητα των Περιορισμών με τίτλο «Restrictions» και την ενότητα των Σχολίων με τίτλο «Annotations». Η ενότητα των Σχολίων περιέχει τα Σχόλια που έχουν οριστεί για την συγκεκριμένη ODM Κλάση και έχει τη μορφή της εικόνας 51. Η ενότητα των Περιορισμών περιέχει τους Περιορισμούς (Restrictions) που έχουν οριστεί για την συγκεκριμένη ODM Κλάση. Τα περιεχόμενα της ενότητας αυτής φαίνονται στην εικόνα 54.




**Εικόνα 54: Ενότητα Περιορισμών Πίνακα Ιδιοτήτων ODM Κλάσης**

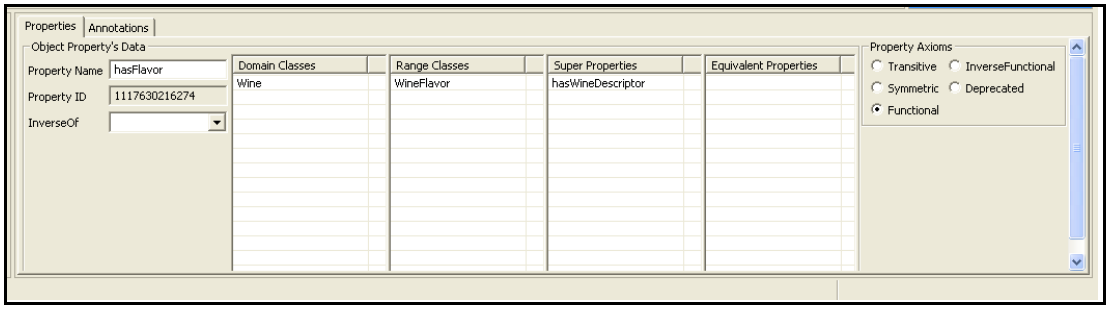
Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποια Ιδιότητα τύπου «Datatype Property» φαίνονται στην εικόνα 55.




**Εικόνα 55: Πίνακας Ιδιοτήτων των Ιδιοτήτων τύπου «Datatype Property»**

Η ενότητα (tab) των Σχολίων με τίτλο «Annotations» του Πίνακα Ιδιοτήτων της εικόνας 55 περιέχει τα Σχόλια που έχουν οριστεί για τη συγκεκριμένη Ιδιότητα τύπου «Datatype Property» και έχουν τη μορφή της εικόνας 51.

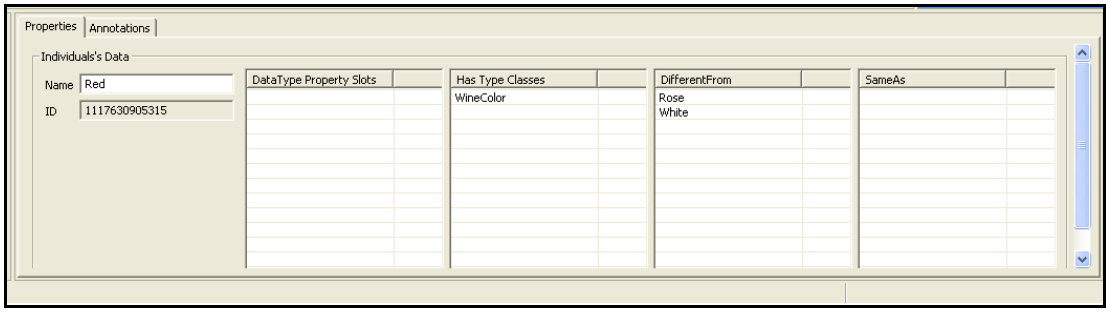
Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποια Ιδιότητα τύπου «Object Property» φαίνονται στην εικόνα 56.




**Εικόνα 56: Πίνακας Ιδιοτήτων των Ιδιοτήτων τύπου «Object Property»**

Η ενότητα (tab) των Σχολίων με τίτλο «Annotations» του Πίνακα Ιδιοτήτων της εικόνας 56 περιέχει τα Σχόλια που έχουν οριστεί για τη συγκεκριμένη Ιδιότητα τύπου «Object Property» και έχουν τη μορφή της εικόνας 51.

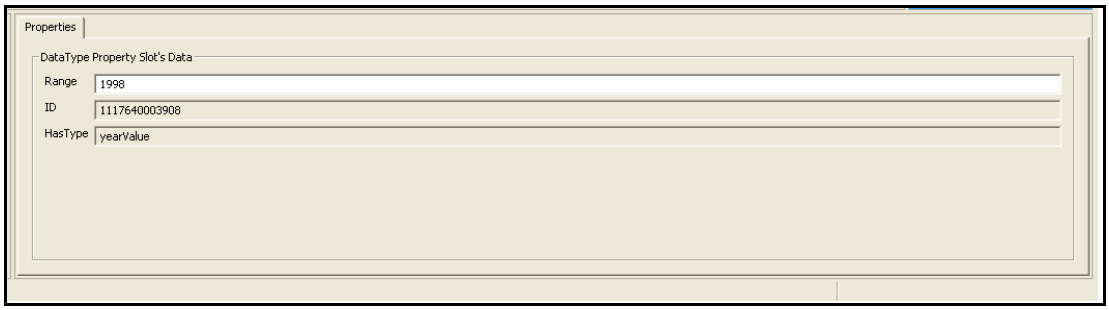
Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποιο Στιγμιότυπο Κλάσης (ClassThing) φαίνονται στην εικόνα 57.




**Εικόνα 57: Πίνακας Ιδιοτήτων Στιγμιότυπου Κλάσης**

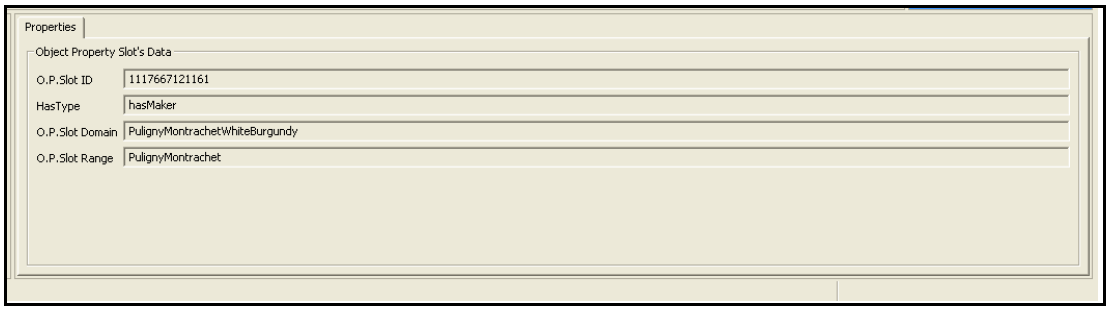
Η ενότητα (tab) των Σχολίων με τίτλο «Annotations» του Πίνακα Ιδιοτήτων της εικόνας 57 περιέχει τα Σχόλια που έχουν οριστεί για το συγκεκριμένη Στιγμιότυπο Κλάσης και έχουν τη μορφή της εικόνας 51.

Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποιο Στιγμιότυπο Ιδιότητας τύπου «Datatype Property» (DatatypePropertyThing) φαίνονται στην εικόνα 58.



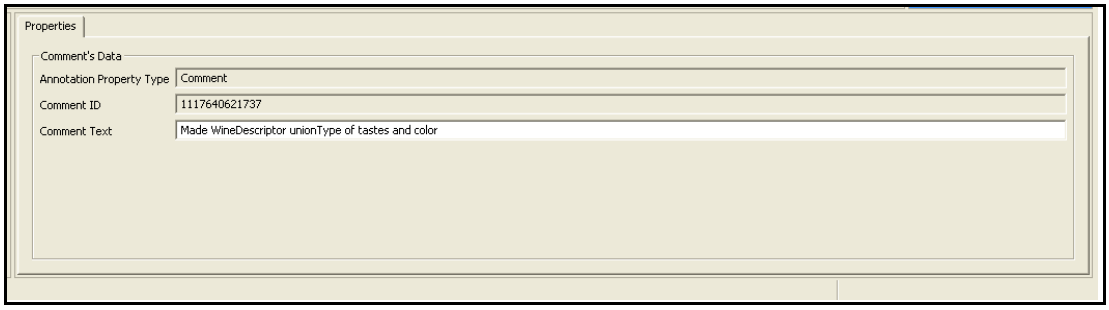

**Εικόνα 58: Πίνακας Ιδιοτήτων Στιγμιότυπου Ιδιότητας τύπου «Datatype Property»**

Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποιο Στιγμιότυπο Ιδιότητας τύπου «Object Property» (ObjectPropertyThing) φαίνονται στην εικόνα 59.



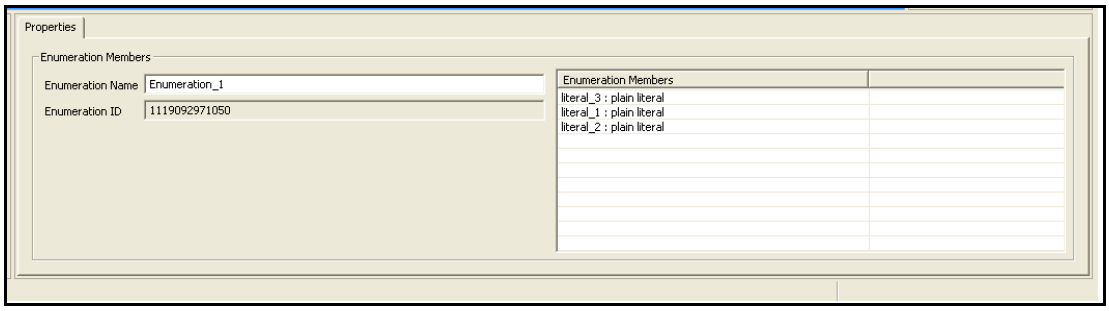

**Εικόνα 59: Πίνακας Ιδιοτήτων Στιγμιότυπου Ιδιότητας τύπου «Object Property»**

Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποιο Σχόλιο τύπου «Comment» φαίνονται στην εικόνα 60.



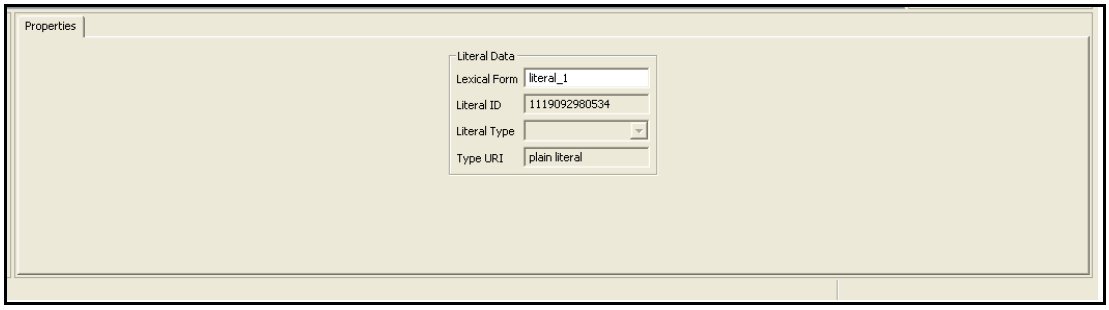

**Εικόνα 60: Πίνακας Ιδιοτήτων Σχολίου τύπου «Comment»**

Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποια Απαρίθμηση (Enumeration) φαίνονται στην εικόνα 61.




**Εικόνα 61: Πίνακας Ιδιοτήτων Απαρίθμησης**

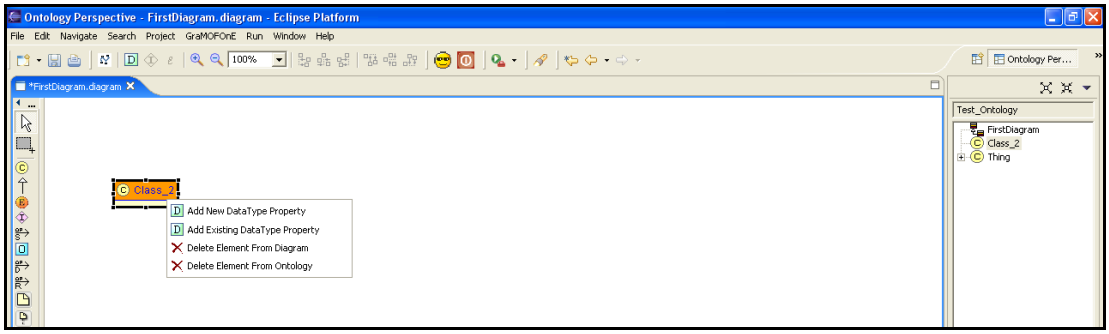
Τα περιεχόμενα του Πίνακα Ιδιοτήτων όταν επιλεγεί κάποιο «Plain Literal» φαίνονται στην εικόνα 62.




**Εικόνα 62: Πίνακας Ιδιοτήτων «Plain Literal»**

### Context Μενού (Menu) Διαγραμμάτων

Το GRAMOFONE υποστηρίζει την ύπαρξη και την λειτουργία ενός context μενού (menu), το οποίο ενεργοποιείται από τον χρήστη κάνοντας δεξί κλικ σε κάποιο από τα γραφικά στοιχεία που βρίσκονται σε ένα διάγραμμα. Οι δυνατές επιλογές που εμφανίζονται από το context μενού εξαρτώνται και είναι πλήρως προσαρμοσμένες στο γραφικό στοιχείο που έχει επιλεγεί, το οποίο σημαίνει πως κάθε γραφικό στοιχείο εμφανίζει και διαφορετικό context μενού. Το context μενού έχει την μορφή της εικόνας 63, στην οποία απεικονίζεται το context μενού μίας ODM Κλάσης.




**Εικόνα 63: Παράδειγμα Context Μενού**

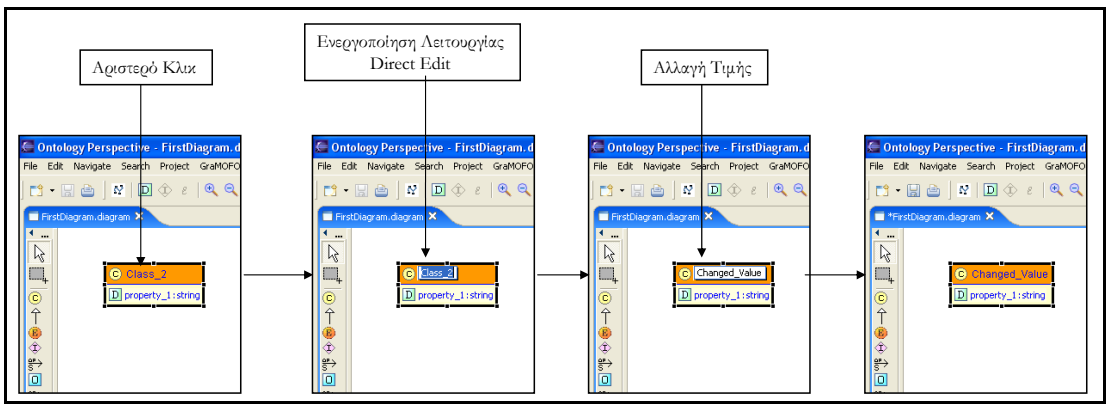
Οι πιθανές επιλογές του context μενού ανάλογα με το επιλεγμένο γραφικό στοιχείο συνοψίζονται στον παρακάτω πίνακα.

| Επιλεγμένο Στοιχείο / Λειτουργία | Προσθήκη Ιδιότητας Datatype Property | Προσθήκη Υπάρχουσας Ιδιότητας Datatype Property | Προσθήκη Στιγμιότυπου Ιδιότητας Datatype Property | Προσθήκη Plain Literal | Διαγραφή Ιδιότητας Datatype Property Από Κλάση | Διαγραφή Στοιχείου Από Διάγραμμα | Διαγραφή Στοιχείου Από Οντολογία |
|---|---|---|---|---|---|---|---|
| Κλάση | ΝΑΙ | ΝΑΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |
| Απαρίθμηση | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |
| Στιγμιότυπο Κλάσης | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |
| Στιγμιότυπο Ιδιότητας Object Property | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |
| Ιδιότητα Object Property | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |
| Σχόλιο Τύπου «Comment» | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |
| Ιδιότητα Datatype Property | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΟΧΙ | ΝΑΙ |
| Στιγμιότυπο Ιδιότητας Datatype Property | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ |
| Plain Literal | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ |
| Συνδέσεις | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΟΧΙ | ΝΑΙ | ΝΑΙ |

**Εικόνα 64: Επιλογές Context Μενού**

### Άμεση Επεξεργασία (Direct Edit)

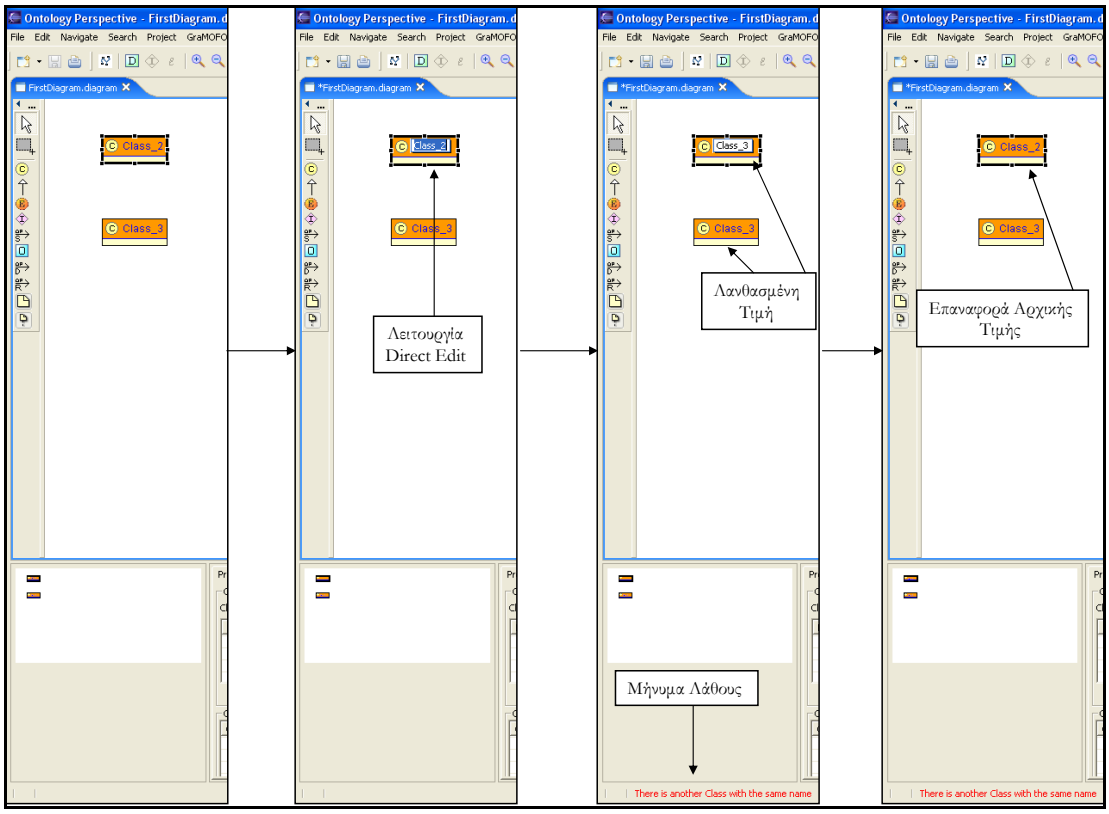
Η λειτουργία Άμεσης Επεξεργασίας (από εδώ και στο εξής «direct edit») παρέχεται από το πλαίσιο εργασίας GEF. Η λειτουργία δίνει τη δυνατότητα της γραφικής (άμεσης) επεξεργασίας των γραφικών στοιχείων ενός διαγράμματος. Με τον τρόπο αυτό είναι δυνατή η γραφική τροποποίηση των ιδιοτήτων των όρων που φαίνονται γραφικά για κάθε έννοια του ODM. Η λειτουργία «direct edit» ενεργοποιείται όταν ο χρήστης κάνει αριστερό κλικ στην ιδιότητα της γραφικής αναπαράστασης του όρου που επιθυμεί να τροποποιήσει. Ένα παράδειγμα της χρήσης της λειτουργίας «direct edit» φαίνεται στην εικόνα 65. Στην εικόνα αυτή τροποποιείται το όνομα (name) μίας ODM Κλάσης μέσω του «direct edit».



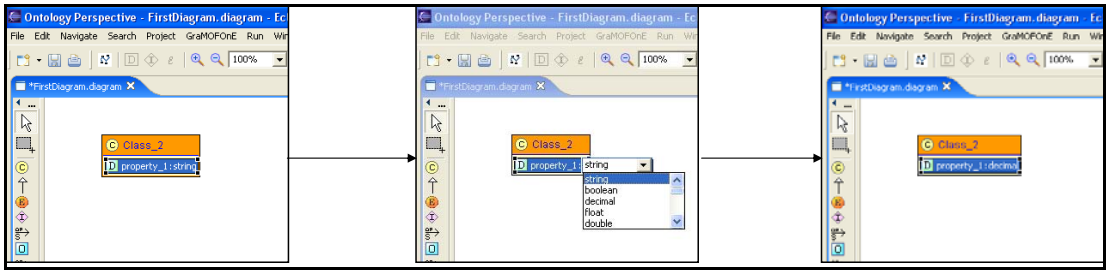

**Εικόνα 65: Παράδειγμα Άμεσης Επεξεργασίας (Direct Edit)**

Η λειτουργία «direct edit» είναι διαθέσιμη για τις γραφικές αναπαραστάσεις των παρακάτω ODM εννοιών:

- ODM Κλάση, Απαρίθμηση (Enumeration), Στιγμιότυπο Κλάσης (ClassThing) και Ιδιότητα τύπου «Object Property»: Δίνεται η δυνατότητα τροποποίησης του ονόματος (name) των εννοιών αυτών.
- Σχόλιο (Annotation) τύπου «Comment»: Δίνεται η δυνατότητα τροποποίησης του κειμένου του σχολίου, το οποίο φαίνεται γραφικά.
- Και στις δύο παραπάνω περιπτώσεις κάθε ενέργεια «direct edit» συσχετίζεται με έναν μηχανισμό επικύρωσης (validator), ο οποίος ελέγχει τις τιμές που εισάγει ο χρήστης. Ο μηχανισμός αυτός απαγορεύει κάποιες συγκεκριμένες τιμές και εμφανίζει στη μπάρα μηνυμάτων (status bar) του GRAMOFONE μηνύματα που ενημερώνουν τον χρήστη για το λάθος. Ένα παράδειγμα επέμβασης του μηχανισμού αυτού φαίνεται στην εικόνα 66. Οι λανθασμένες τιμές που απαγορεύονται είναι οι εξής:
    - Όταν το πεδίο δεν έχει καθόλου τιμή (null).
    - Όταν εισάγεται ο χαρακτήρας " " (space).
    - Απαγορεύει την ύπαρξη εννοιών με το ίδιο όνομα.
- Επιπλέον στοιχεία που μπορούν να τροποποιηθούν μέσω «direct edit» είναι οι Ιδιότητες τύπου «Datatype Property», η έννοια «Plain Literal» και τα Στιγμιότυπα Ιδιοτήτων τύπου «Datatype Property» (DatatypePropertyThings). Στις Ιδιότητες τύπου «Datatype Property» τροποποιείται το όνομά τους (name), στα «Plain Literals» τροποποιείται η ιδιότητα «lexicalForm», και στα Στιγμιότυπα Ιδιοτήτων τύπου «Datatype Property» (DatatypePropertyThings) η ιδιότητα «range». Κάθε ενέργεια «direct edit» συσχετίζεται με ένα μηχανισμό επικύρωσης (validator), ο οποίος ελέγχει τις τιμές που εισάγει ο χρήστης. Ο μηχανισμός αυτός απαγορεύει κάποιες συγκεκριμένες τιμές και εμφανίζει στη μπάρα μηνυμάτων (status bar) του GRAMOFONE μηνύματα που ενημερώνουν τον χρήστη για το λάθος. Οι λανθασμένες τιμές που απαγορεύονται είναι οι εξής:
    - Όταν το πεδίο δεν έχει καθόλου τιμή (null).
    - Όταν εισάγεται ο χαρακτήρας " " (space).
    - Απαγορεύει Ιδιότητες τύπου «Datatype Property» με το ίδιο όνομα.
- Οι Ιδιότητες τύπου «Datatype Property» εκτός από το όνομά τους (name), απεικονίζουν γραφικά και το εύρος τιμών τους (DataRange). Αυτή η ιδιότητα μπορεί να τροποποιηθεί μέσω «direct edit» με την χρήση ενός μηχανισμού (combo box), όπως φαίνεται στην εικόνα 67.



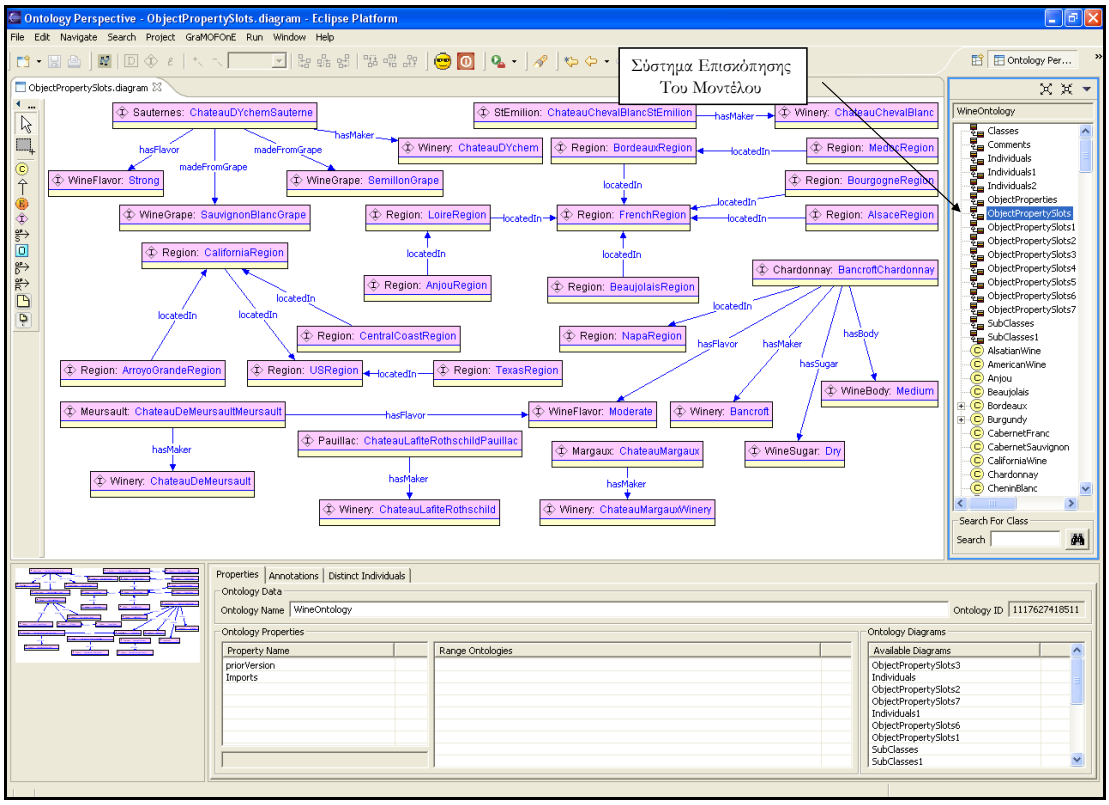

**Εικόνα 66: Μηχανισμός Επικύρωσης (Validator) για «Direct Edit»**



**Εικόνα 67: Μηχανισμός (Combo Box) Αλλαγής Εύρους Τιμών μέσω «Direct Edit» για Ιδιότητες τύπου «Datatype Property»**

### Σύστημα Επισκόπησης του Μοντέλου (Model Explorer)

Το Σύστημα Επισκόπησης του Μοντέλου (Model Explorer), το οποίο φαίνεται στην εικόνα 68, αποτελεί την υλοποίηση της αντίστοιχης υπό-μονάδας που αναλύθηκε στο κεφάλαιο της αρχιτεκτονικής του GRAMOFONE (κεφάλαιο 6). Η λειτουργικότητά του περιλαμβάνει τόσο την πλήρη αναπαράσταση της οντολογίας που δημιουργεί ο χρήστης με μορφή δέντρου, όσο και την ιδιότητα που έχει να λειτουργεί σαν πηγή (source) εννοιών που έχουν οριστεί στην οντολογία και οι οποίες μέσω Drag And Drop (DND) μπορούν να επαναχρησιμοποιηθούν σε άλλα διαγράμματα χωρίς να χρειάζεται να επαναπροσδιοριστούν. Η λειτουργία του DND θα αναλυθεί στην συνέχεια.




**Εικόνα 68: Σύστημα Επισκόπησης του Μοντέλου**

Το Σύστημα Επισκόπησης του Μοντέλου (Model Explorer) έχει την ιδιότητα όποτε επιλέγεται κάποιο στοιχείο σε ένα διάγραμμα να επιλέγεται αυτόματα σε αυτό (στον Model Explorer) το επιλεγμένο γραφικά στοιχείο (να γίνεται δηλαδή highlighted).

Το βασικό δομικό στοιχείο του Συστήματος Επισκόπησης του Μοντέλου είναι ένας JFace δενδρικός μηχανισμός επισκόπησης (TreeViewer) ο οποίος χρησιμοποιείται για την αναπαράσταση των οντολογιών.

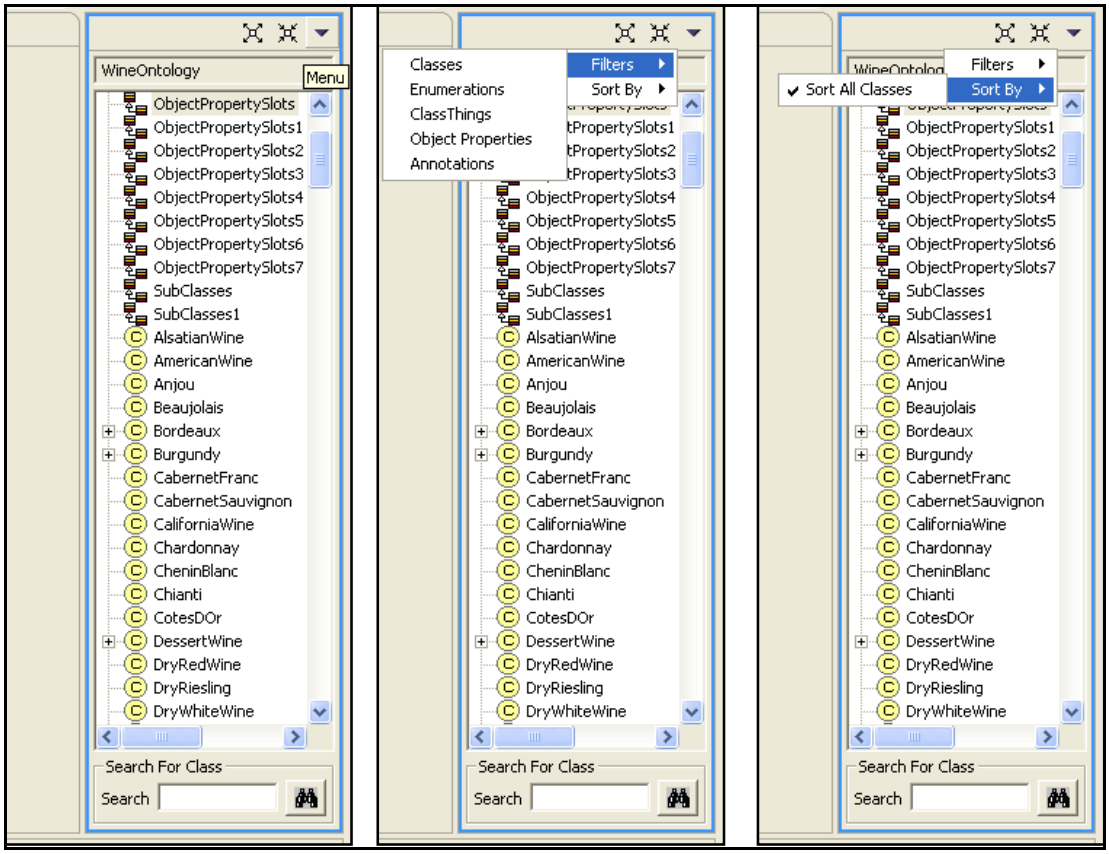
### Λειτουργικότητα του Συστήματος Επισκόπησης του Μοντέλου (Model Explorer)

Στην ενότητα θα παρουσιαστεί η λειτουργικότητα που παρέχεται από το Σύστημα Επισκόπησης του Μοντέλου.

Το Σύστημα Επισκόπησης του Μοντέλου περιέχει μία μπάρα εργαλείων (tool bar) και ένα μενού (menu), τα οποία περιέχουν μέρος της λειτουργικότητας του Συστήματος Επισκόπησης του Μοντέλου. Το μενού με τις δυνατές επιλογές του φαίνεται στην εικόνα 69. Όπως φαίνεται και στην εικόνα το μενού παρέχει δύο υπό-μενού, ένα για το φιλτράρισμα των στοιχείων που φαίνονται στο Σύστημα Επισκόπησης του Μοντέλου και ένα για την ταξινόμηση ή την μη ταξινόμηση των στοιχείων αυτών.

Το μενού φιλτραρίσματος δίνει στο χρήστη την δυνατότητα να επιλέξει το είδος των στοιχείων που φαίνονται στο Σύστημα Επισκόπησης του Μοντέλου. Επιτρέπει ακόμα και τον συνδυασμό των φίλτρων δηλαδή την εμφάνιση δύο ή περισσοτέρων ειδών στοιχείων. Οι

δυνατές επιλογές φιλτραρίσματος γίνονται μεταξύ των εννοιών Κλάσης, Απαρίθμησης (Enumeration), Στιγμιότυπου Κλάσης (ClassThing), Ιδιότητας τύπου «Object Property» και Σχολίου (Annotation) τύπου «Comment». Τα διαγράμματα της οντολογίας φαίνονται πάντα στο Σύστημα Επισκόπησης του Μοντέλου. Το μενού ταξινόμησης δίνει την δυνατότητα στον χρήστη να επιλέξει αν τα στοιχεία του Συστήματος Επισκόπησης του Μοντέλου θα εμφανίζονται ταξινομημένα κατά είδος και κατά αλφαβητική σειρά ή όχι. Η προεπιλεγμένη (default) επιλογή του Συστήματος Επισκόπησης του Μοντέλου είναι τα στοιχεία να εμφανίζονται ταξινομημένα.

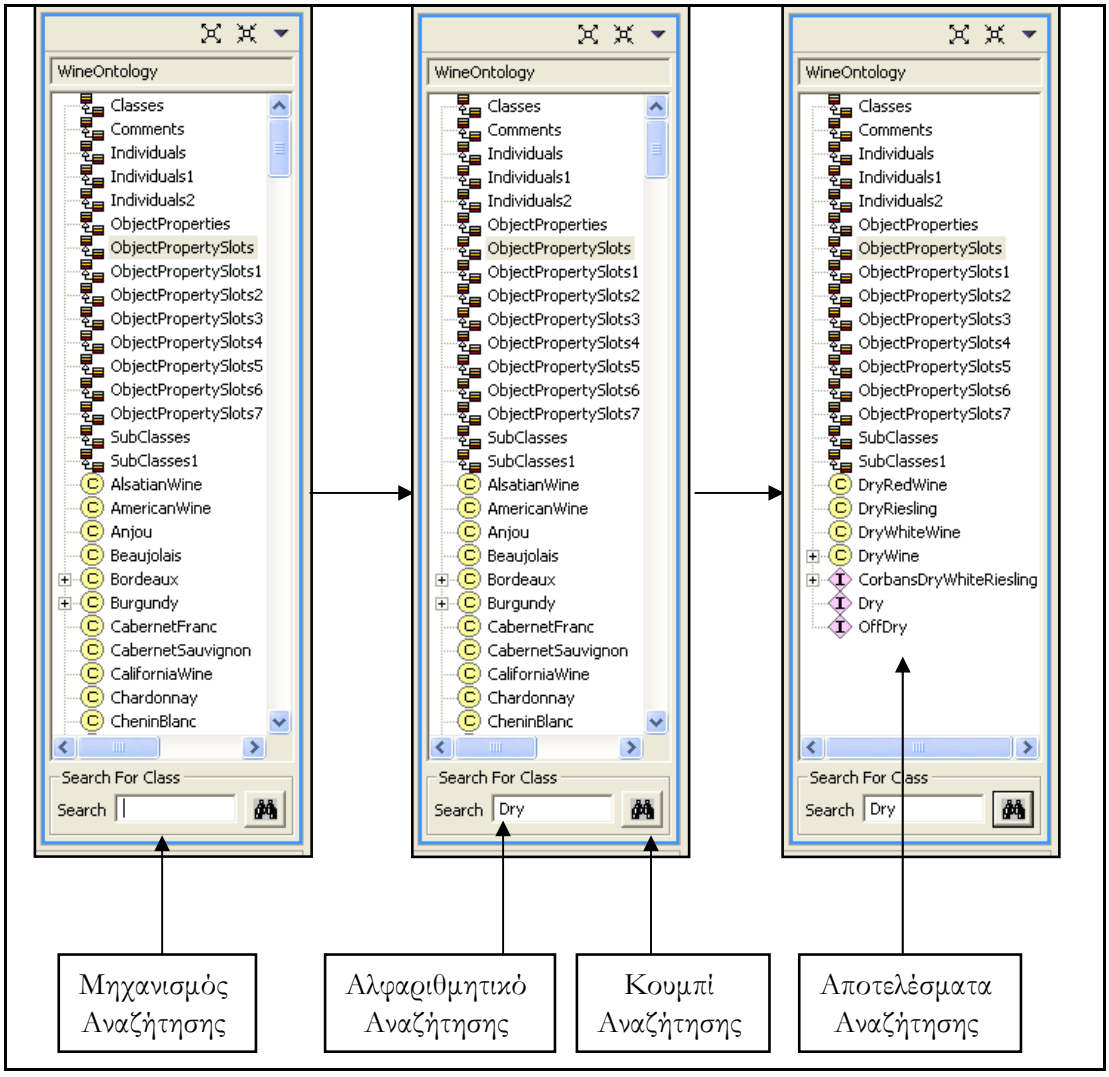



**Εικόνα 69: Το Μενού του Συστήματος Επισκόπησης του Μοντέλου**

### Μηχανισμός Αναζήτησης Στοιχείων της Οντολογίας

Το Σύστημα Επισκόπησης του Μοντέλου δίνει ακόμα την δυνατότητα αναζήτησης στοιχείων που περιέχονται στην οντολογία. Για το σκοπό αυτό υλοποιήθηκε ο μηχανισμός αναζήτησης του Συστήματος Επισκόπησης του Μοντέλου το οποίο φαίνεται στην εικόνα 70. Το κριτήριο αναζήτησης είναι ένα αλφαριθμητικό (string) το οποίο εισάγεται από το χρήστη στη φόρμα (text box) του μηχανισμού αναζήτησης. Το Σύστημα Επισκόπησης του Μοντέλου στη συνέχεια παρουσιάζει όλα τα στοιχεία της οντολογίας τα οποία περιέχουν στον τίτλο τους το αλφαριθμητικό (string) που εισήγαγε ο χρήστης. Ένα παράδειγμα αναζήτησης φαίνεται στην εικόνα 70. Το αλφαριθμητικό που χρησιμοποιήθηκε σαν κριτήριο

αναζήτησης είναι το "Dry". Παρατηρούμε πως το Σύστημα Επισκόπησης του Μοντέλου ανανέωσε το περιεχόμενό του ώστε να παρουσιάζει (πέρα από τα διαγράμματα τα οποία φαίνονται πάντα) τα στοιχεία της οντολογίας τα οποία περιέχουν στον τίτλο τους το αλφαριθμητικό "Dry". Να σημειωθεί πως η αναζήτηση γίνεται μεταξύ των εννοιών Κλάσεων, Απαριθμήσεων (Enumerations), Στιγμιότυπων Κλάσεων (ClassThings), Ιδιοτήτων τύπου «Object Property» και Σχολίων (Annotations) τύπου «Comment» καθώς οι υπόλοιπες έννοιες "περιέχονται" κατά κάποιο τρόπο στις προαναφερόμενες.

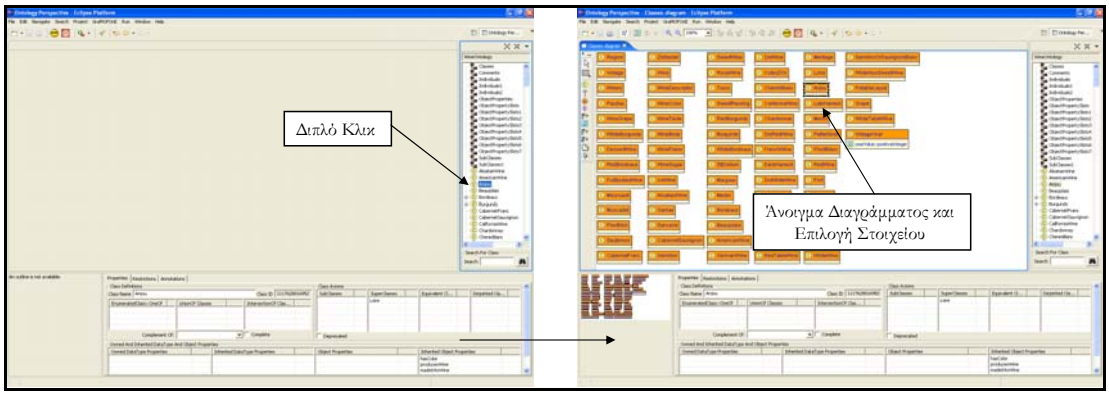



**Εικόνα 70: Μηχανισμός Αναζήτησης Στοιχείων της Οντολογίας**

### Υποστήριξη Άμεσης Επιλογής Γραφικών Στοιχείων

Το Σύστημα Επισκόπησης του Μοντέλου έχει την ιδιότητα, όταν ο χρήστης κάνει διπλό κλικ πάνω σε κάποιο από τα στοιχεία του JFace δενδρικού μηχανισμού επισκόπησης (TreeViewer) του Συστήματος Επισκόπησης του Μοντέλου, να ανοίγει ένα από τα διαγράμματα στο οποίο περιέχεται το στοιχείο αυτό και να το επιλέγει (highlight) πάνω στο διάγραμμα. Πρόκειται για μια πολύ ενδιαφέρουσα λειτουργία καθώς δίνει τη δυνατότητα

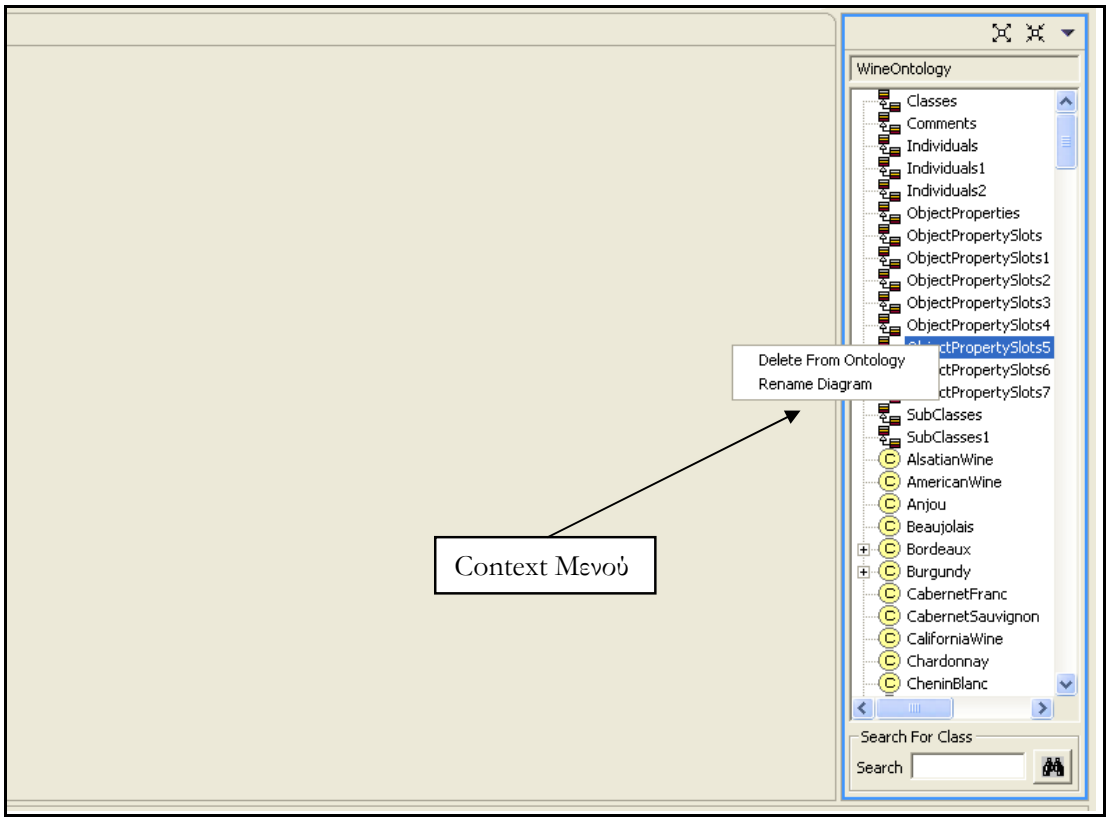
στο χρήστη να βρίσκει σε πιο διάγραμμα έχει ορίσει το στοιχείο που τον ενδιαφέρει όσο μεγάλη και αν είναι η οντολογία που επεξεργάζεται. Ένα παράδειγμα αυτής της ιδιότητας φαίνεται στην εικόνα 71.




**Εικόνα 71: Υποστήριξη Άμεσης Επιλογής Γραφικών Στοιχείων**

### Context Μενού του Συστήματος Επισκόπησης του Μοντέλου

Το Σύστημα Επισκόπησης του Μοντέλου διαθέτει επιπλέον ένα context μενού το οποίο περιέχει την λειτουργία της διαγραφής του στοιχείου που επιλέχθηκε από την οντολογία. Επιπλέον για τα διαγράμματα προστίθεται και η λειτουργία της μετονομασίας. Ένα στιγμιότυπο (screenshot) της λειτουργίας αυτής φαίνεται στην εικόνα 72.



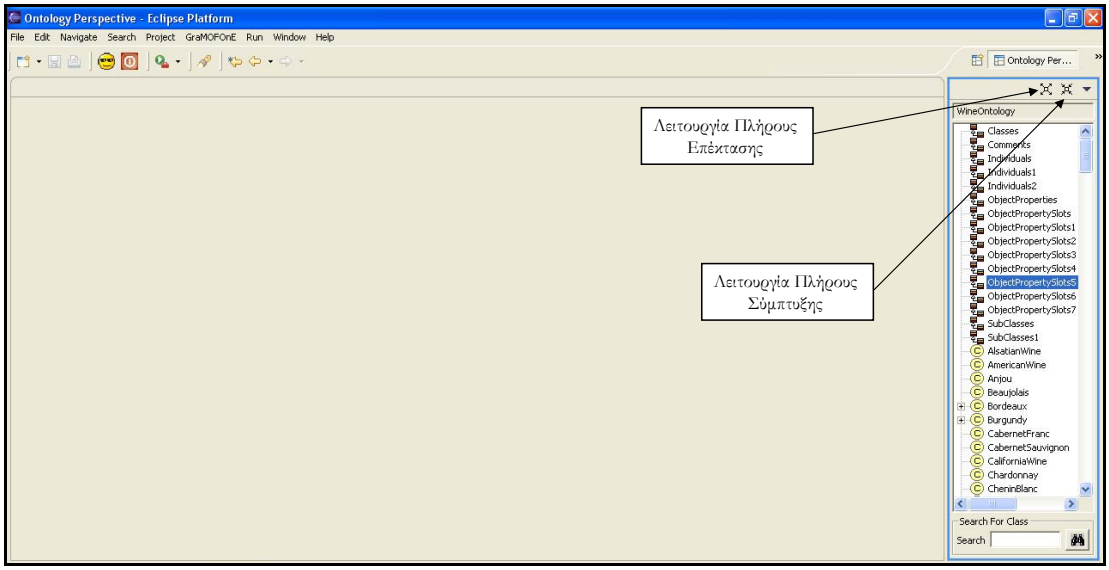

**Εικόνα 72: Context Μενού του Συστήματος Επισκόπησης του Μοντέλου**

**Μπάρα Εργαλείων (Tool Bar) του Συστήματος Επισκόπησης του Μοντέλου**

Το Σύστημα Επισκόπησης του Μοντέλου περιέχει μία μπάρα εργαλείων η οποία περιλαμβάνει δύο λειτουργίες. Οι λειτουργίες αυτές είναι:

- Λειτουργία πλήρους επέκτασης (expand) των στοιχείων του JFace δενδρικού μηχανισμού επισκόπησης (TreeViewer) του Συστήματος Επισκόπησης του Μοντέλου.
- Λειτουργία πλήρους σύμπτυξης (collapse) των στοιχείων του JFace δενδρικού μηχανισμού επισκόπησης (TreeViewer) του Συστήματος Επισκόπησης του Μοντέλου.

Οι λειτουργίες που αναφέρθηκαν πιο πάνω φαίνονται στην εικόνα 73.

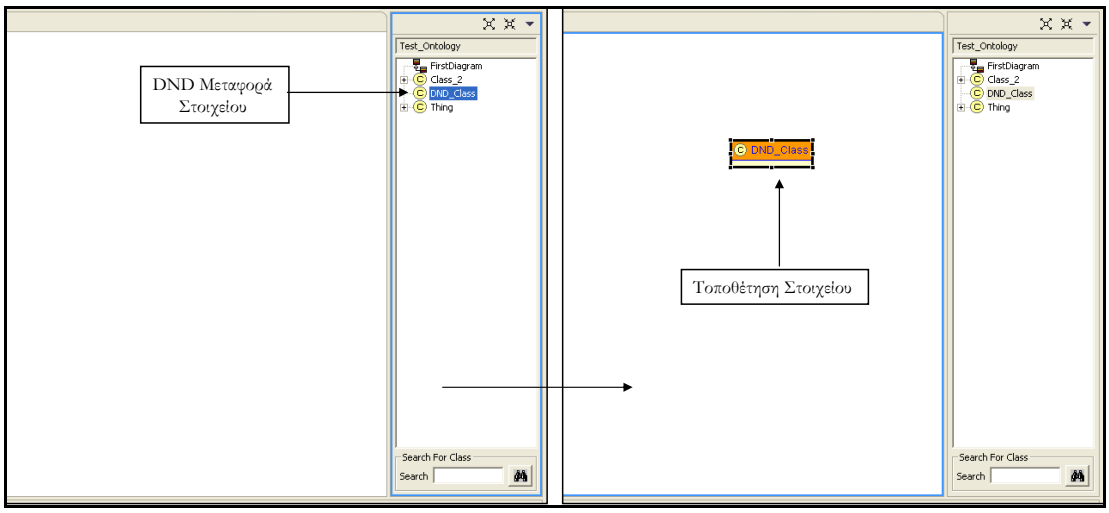



**Εικόνα 73: Μπάρα Εργαλείων του Συστήματος Επισκόπησης του Μοντέλου**

**Λειτουργία Drag And Drop (DND)**

Το GRAMOFONE έχει την ιδιότητα να επιτρέπει την μεταφορά στοιχείων που βρίσκονται στο Σύστημα Επισκόπησης του Μοντέλου (Model Explorer), μέσα σε κάποιο ανοιχτό διάγραμμα. Με τον τρόπο αυτό γίνεται δυνατή η επαναχρησιμοποίηση όρων που έχουν οριστεί στην οντολογία σε παραπάνω από ένα διαγράμματα! Η αξία της λειτουργίας αυτής είναι προφανής και συντελεί σε μεγάλο βαθμό στην διευκόλυνση του χρήστη στην δημιουργία οντολογιών καθώς του δίνεται η δυνατότητα να "σπάσει" τον ορισμό ενός στοιχείου της οντολογίας σε παραπάνω από ένα διαγράμματα. Για παράδειγμα μία ODM Κλάση μπορεί να οριστεί σε δύο διαγράμματα, στο πρώτο να οριστούν οι υπό-κλάσεις της και στο δεύτερο οι Ιδιότητες τύπου «Object Property» στις οποίες συμμετέχει. Οι όροι ο οποίοι μπορούν να μεταφερθούν από το Σύστημα Επισκόπησης του Μοντέλου μέσω Drag And Drop είναι οι Κλάσεις, οι Απαριθμήσεις (Enumerations), τα Στιγμιότυπα Κλάσεων (ClassThings), οι Ιδιότητες τύπου «Object Property», και τα Σχόλια (Annotations) τύπου

«Comment». Επιπλέον όταν κάποιο από τα στοιχεία που ο χρήστης επιχειρεί να μεταφέρει, περιέχεται ήδη στο διάγραμμα που στοχεύει, τότε το DND αποτρέπεται από το GRAMOFONE. Ένα παράδειγμα χρήσης του DND φαίνεται στην εικόνα 74 στην οποία μεταφέρεται μία ODM Κλάση από το Σύστημα Επισκόπησης του Μοντέλου σε κάποιο διάγραμμα.




**Εικόνα 74: Παράδειγμα Drag And Drop**

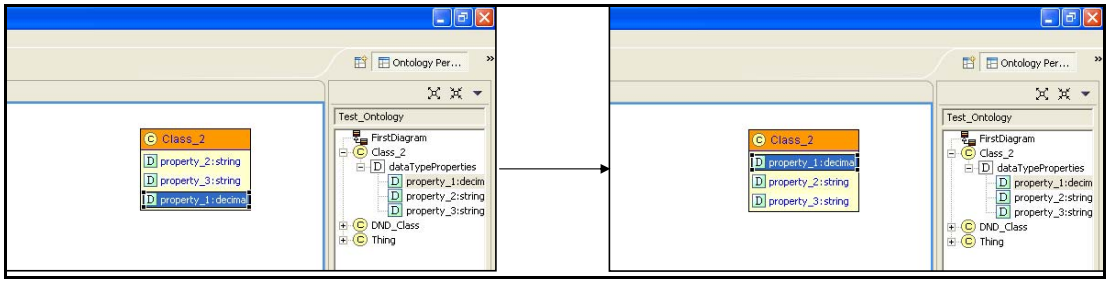
### Συμπληρωματικές Λειτουργίες

Όπως προαναφέρθηκε το GRAMOFONE υποστηρίζει τη δημιουργία και την ανάκτηση πολλαπλών διαγραμμάτων στο πλαίσιο μίας οντολογίας, ιδιότητα εξαιρετικά πολύτιμη για την υλοποίηση μεγάλων οντολογιών. Ενδεικτικά αναφέρεται ότι η υλοποίηση της πρότυπης οντολογίας (wine ontology) που έγινε για τους σκοπούς της αξιολόγησης τόσο του ODM όσο και του GRAMOFONE (θα περιγραφεί στο επόμενο κεφάλαιο) απαίτησε 16 διαγράμματα!

Το GEF παρέχει ένα μηχανισμό με την ονομασία «Key Handlers». Ο μηχανισμός αυτός δίνει την δυνατότητα στο χρήστη να αντιστοιχίσει μία λειτουργία σε κάποιο πλήκτρο έτσι ώστε με το πάτημα του πλήκτρου αυτού να ενεργοποιείται η λειτουργία αυτή. Είναι ουσιαστικά ένας μηχανισμός που δημιουργεί συντομεύσεις (shortcuts). Σαν παράδειγμα χρήσης αυτού του μηχανισμού στο πλαίσιο του GRAMOFONE έχει οριστεί πως με το πάτημα του κουμπιού F2 εμφανίζεται το διάλογος εκτύπωσης της εφαρμογής.

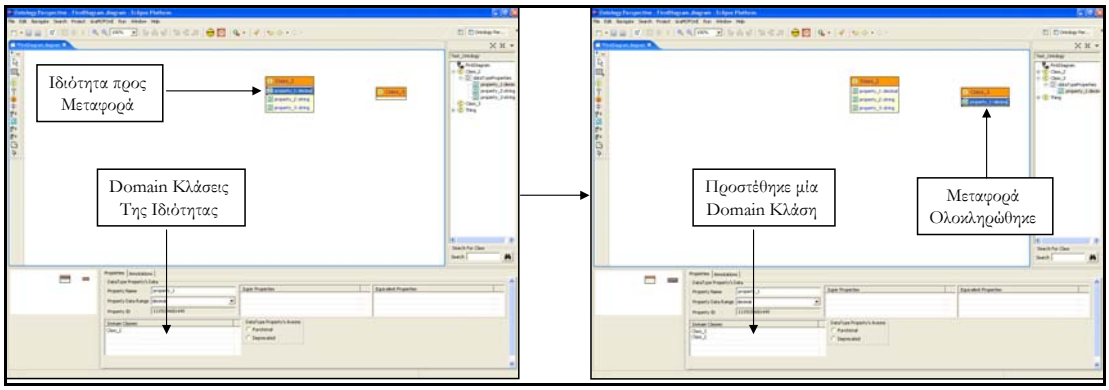
Μία επίσης πολύ χρήσιμη δυνατότητα που παρέχεται από το GRAMOFONE, όσον αφορά τη δημιουργία διαγραμμάτων, είναι η δυνατότητα της μετακίνησης των γραφικών στοιχείων των διαγραμμάτων με τα βέλη του πληκτρολογίου (arrow buttons). Η λειτουργία αυτή ενεργοποιείται με το πάτημα του χαρακτήρα dot (“.”) του πληκτρολογίου όταν έχει επιλεχτεί κάποιο γραφικό στοιχείο σε ένα διάγραμμα.

Επιπλέον το GRAMOFONE παρέχει τη δυνατότητα στο χρήστη να αλλάξει τη σειρά με την οποία φαίνονται γραφικά οι Ιδιότητες τύπου «Datatype Property» μέσα σε μία ODM Κλάση και τα «Plain Literals» τα οποία περιέχονται σε μία Απαρίθμηση (Enumeration). Η αλλαγή της σειράς γίνεται μέσω Drag And Drop (DND). Ένα παράδειγμα χρήσης αυτής της ιδιότητας φαίνεται στην εικόνα 75 στην οποία τροποποιείται η σειρά των Ιδιοτήτων τύπου «Datatype Property» που περιέχονται σε μία ODM Κλάση. Πιο συγκεκριμένα μεταφέρεται η πρώτη Ιδιότητα (Datatype Property) της κλάσης στο τέλος της λίστας των Ιδιοτήτων (Datatype Properties)




**Εικόνα 75: Γραφική Αλλαγή Σειράς Στοιχείων**

Ακόμα το GRAMOFONE παρέχει τη δυνατότητα στο χρήστη να μεταφέρει (copy) Ιδιότητες τύπου «Datatype Property» μεταξύ ODM Κλάσεων και «Plain Literals» μεταξύ Απαριθμήσεων (Enumerations). Αυτή η λειτουργία είναι εξαιρετικά χρήσιμη για την πρόσθεση «Domain» ODM Κλάσεων σε Ιδιότητες τύπου «Datatype Property» και την επαναχρησιμοποίηση «Plain Literals» σε παραπάνω από μία Απαριθμήσεις (Enumerations). Παράδειγμα χρήσης της λειτουργίας αυτής φαίνεται στην εικόνα 76. Στην εικόνα φαίνεται η μεταφορά μίας Ιδιότητας (Datatype Property) από μία ODM Κλάση σε μία άλλη οπότε κατά συνέπεια, η Ιδιότητα (Datatype Property) μετά την ολοκλήρωση της μεταφοράς αποκτά και δεύτερη «Domain» Κλάση.




**Εικόνα 76: Προσθήκη «Domain» Κλάσης σε Ιδιότητα τύπου «Datatype Property» μέσω DND**

**Ανακεφαλαίωση**

Στο κεφάλαιο παρουσιάστηκε το γραφικό περιβάλλον εργασίας (GUI) του GRAMOFONE. Αρχικά περιγράφηκαν οι βασικές αρχές σχεδιασμού οι οποίες λήφθηκαν υπόψη κατά τη διάρκεια του σχεδιασμού του περιβάλλοντος εργασίας του εργαλείου και στη συνέχεια περιγράφηκε πλήρως με τη βοήθεια στιγμιότυπων (screenshots) το περιβάλλον εργασίας του εργαλείου GRAMOFONE. Η περιγραφή του περιβάλλοντος εργασίας περιελάμβανε τις περιγραφές του «perspective» του εργαλείου, της λειτουργικότητας του μενού και της μπάρας εργαλείων του GRAMOFONE, της παλέτας που συνοδεύει τα διαγράμματα, της γραφικής αναπαράστασης των εννοιών της μοντέλου της εφαρμογής (οντολογία), των επιπρόσθετων ιδιοτήτων των διαγραμμάτων, της αναπαράστασης των χαρακτηριστικών των εννοιών του ODM στο εργαλείο, του context μενού των διαγραμμάτων, της λειτουργίας άμεσης επεξεργασίας (direct edit), του Συστήματος Επισκόπησης του Μοντέλου (Model Explorer) και των συμπληρωματικών λειτουργιών που παρέχει το εργαλείο.

Κεφάλαιο 8

# ΑΞΙΟΛΟΓΗΣΗ ΤΟΥ GRAMOFONE ΚΑΙ ΤΟΥ ODM

**Εισαγωγή**

Στο κεφάλαιο θα παρουσιάσουμε τη διαδικασία και τα αποτελέσματα της αξιολόγησης που έγινε τόσο για το μετά-μοντέλο ODM όσο και για το εργαλείο GRAMOFONE. Η αξιολόγηση αυτή έγινε με την ανάπτυξη μιας οντολογίας με τη χρήση του εργαλείου και την αποθήκευση της στη Βάση Γνώσης μέσω των JMI διεπαφών της που υλοποιούν το μετά-μοντέλο ODM. Σκοπός της αξιολόγησης αυτής ήταν να εξεταστεί η πληρότητα και η ορθότητα του μετά-μοντέλου καθώς και η συμβατότητά του με τη γλώσσα OWL. Όσον αφορά το εργαλείο GRAMOFONE, στόχος της αξιολόγησης ήταν να ελεγχθεί η πληρότητά του, η αξιοπιστία του, και η χρηστικότητά του.

Η οντολογία που αναπτύχθηκε είναι μια οντολογία κρασιών (wine ontology) η οποία είναι μία ευρέως χρησιμοποιημένη οντολογία η οποία δίδεται (μέσω του διαδικτυακού τόπου http://protege.stanford.edu/plugins/owl/owl-library/) από το πανεπιστήμιο Stanford ως οντολογία δοκιμών (testing ontology) για εργαλεία σχετικά με οντολογίες (ontology editors, reasoners, etc.). Η οντολογία αυτή περιλαμβάνει όλες τις έννοιες που ορίζονται από το ODM, εκτός από την έννοια της Απαρίθμησης (Enumeration). Το RDF αρχείο που περιέχει την οντολογία παρέχεται στο Παράρτημα Α.

Για την υλοποίηση της οντολογίας δημιουργήθηκαν δεκαέξι διαγράμματα. Τα διαγράμματα και το είδος των εννοιών που περιέχονται σε κάθε διάγραμμα θα παρουσιαστούν στην επόμενη ενότητα. Κατόπιν θα περιγράψουμε τα αποτελέσματα της αξιολόγησης αναφέροντας πλεονεκτήματα και μειονεκτήματα που διαπιστώθηκαν. Επιπλέον θα παρουσιαστούν οι παραδοχές που έγιναν στο πλαίσιο του GRAMOFONE οι οποίες θέτουν κάποιες μικρές διαφοροποιήσεις του μοντέλου της εφαρμογής από αυτό που ορίζεται στο ODM.

**Υλοποίηση Πρότυπης Οντολογίας (Wine Ontology)**

Η υλοποίηση της οντολογίας κρασιών χωρίστηκε σε διαγράμματα το καθένα από τα οποία ορίζει κάποιες συγκεκριμένες έννοιες της οντολογίας. Οι έννοιες που ορίζονται σε κάθε διάγραμμα φανερώνονται από τον τίτλο του διαγράμματος.

Οι Κλάσεις της οντολογίας ορίστηκαν στο διάγραμμα με τίτλο «Classes». Στο διάγραμμα ορίστηκε και η μοναδική Ιδιότητα τύπου «Datatype Property» της οντολογίας. Το διάγραμμα φαίνεται στην εικόνα 77.

Τα Σχόλια (Annotations) τύπου «Comment» της οντολογίας ορίστηκαν στο διάγραμμα με τίτλο «Comments». Να σημειωθεί πως μόνο αυτό το είδος σχολίων επιτρέπεται να αναπαρασταθεί γραφικά στο GRAMOFONE. Το διάγραμμα φαίνεται στην εικόνα 78.

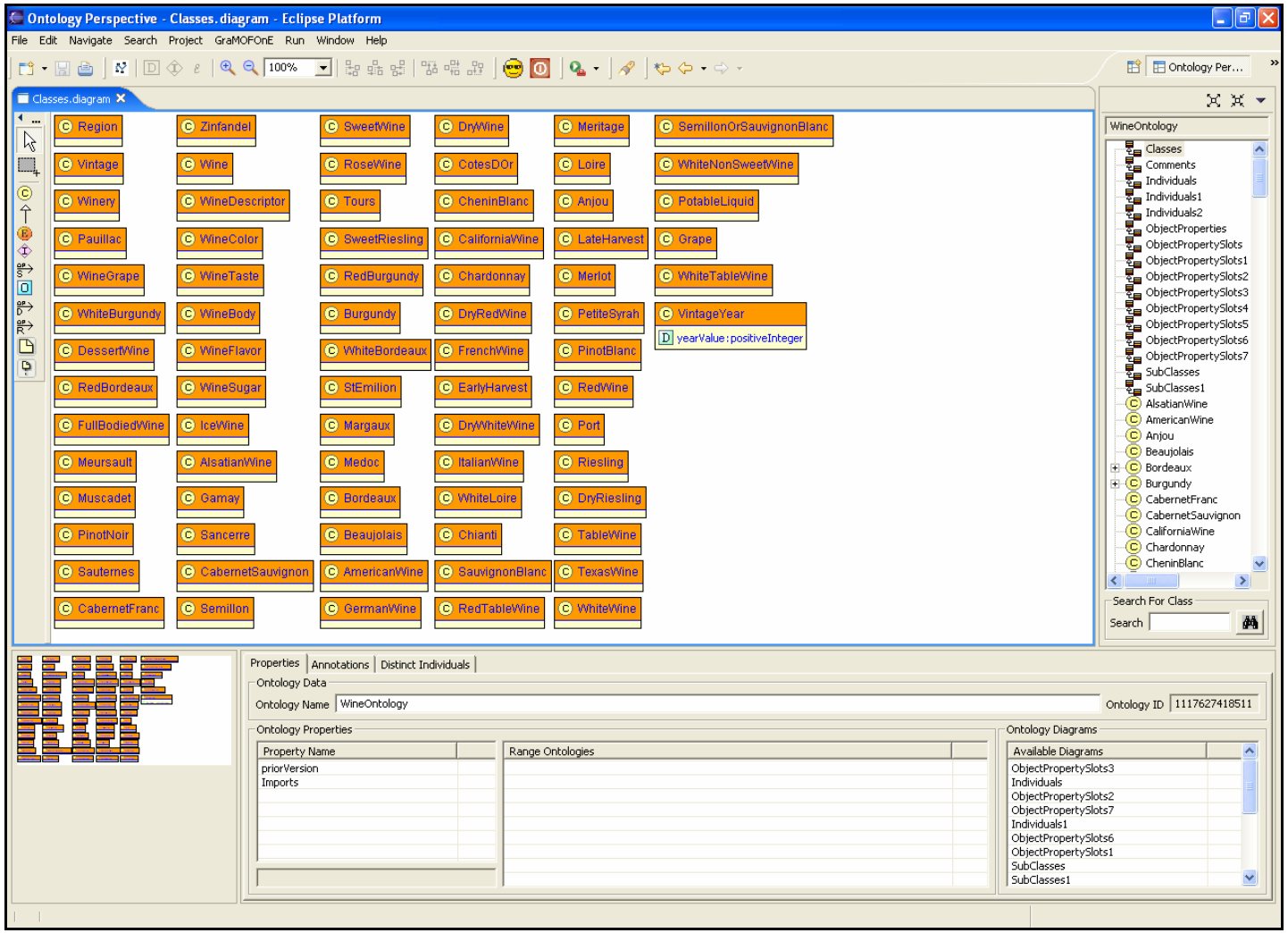
Τα Στιγμιότυπα Κλάσεων (ClassThings) της οντολογίας ορίστηκαν στα διαγράμματα με τίτλους «Individuals», «Individuals1» και «Individuals2». Στο διάγραμμα με τίτλο «Individuals» ορίστηκε και το μοναδικό Στιγμιότυπο Ιδιότητας τύπου «Datatype Property» (DatatypePropertyThing) της οντολογίας. Τα διαγράμματα φαίνονται στις εικόνες 79, 80 και 81 αντίστοιχα.

Οι Ιδιότητες τύπου «Object Property» ορίστηκαν στο διάγραμμα με τίτλο «ObjectProperties». Το διάγραμμα φαίνεται στην εικόνα 82.

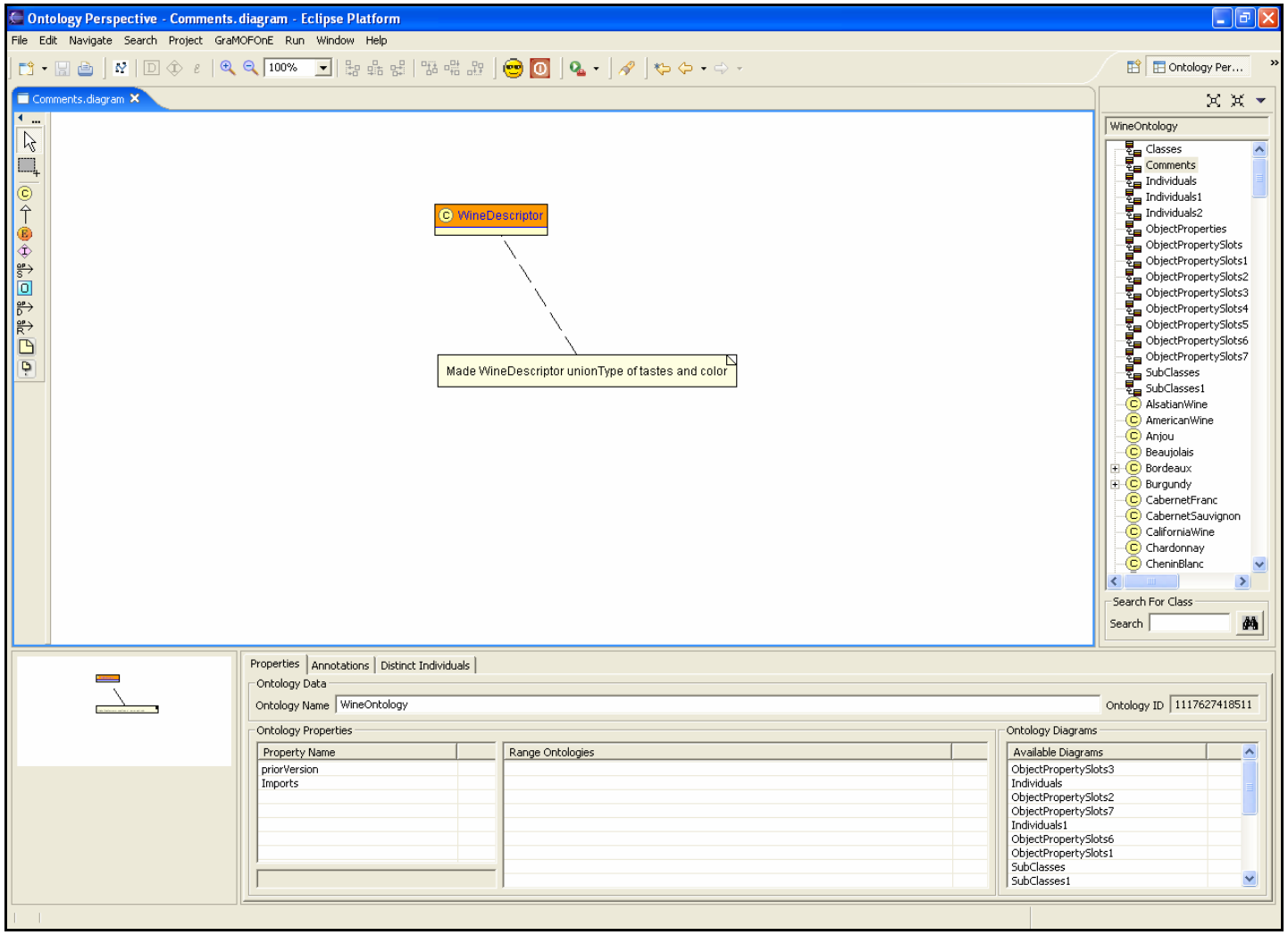
Τα Στιγμιότυπα Ιδιοτήτων τύπου «Object Property» (ObjectPropertyThings) ορίστηκαν στα διαγράμματα με τίτλους «ObjectPropertySlots», «ObjectPropertySlots1», «ObjectPropertySlots2», «ObjectPropertySlots3», «ObjectPropertySlots4», «ObjectPropertySlots5», «ObjectPropertySlots6» και «ObjectPropertySlots7». Τα διαγράμματα φαίνονται στις εικόνες 83, 84, 85, 86, 87, 88, 89 και 90 αντίστοιχα.

Οι Υπό-Κλάσεις της οντολογίας ορίστηκαν στα διαγράμματα με τίτλους «SubClasses» και «SubClasses1». Τα διαγράμματα φαίνονται στις εικόνες 91 και 92 αντίστοιχα.

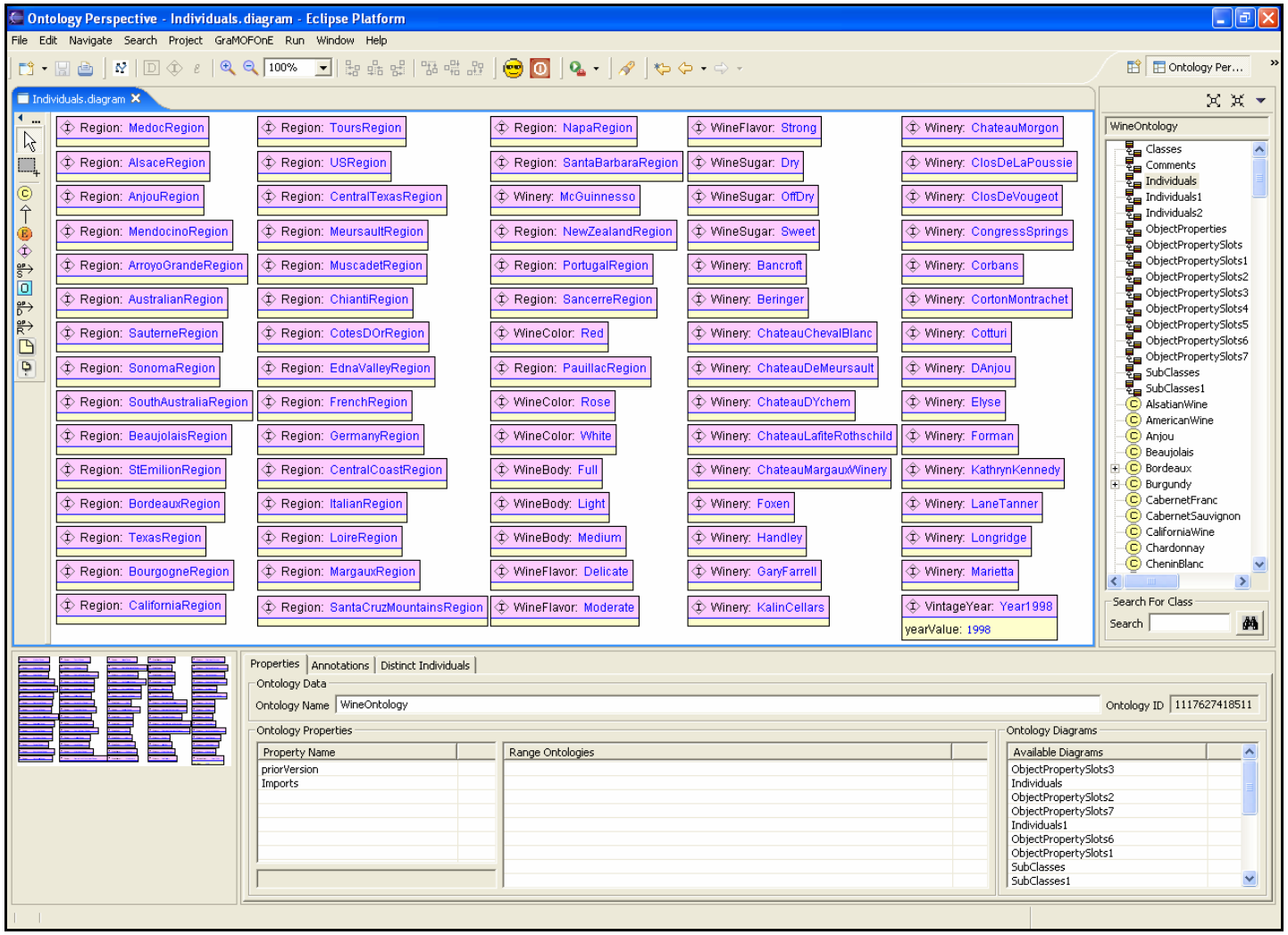
Οι έννοιες οι οποίες δεν έχουν γραφική αναπαράσταση (δεν φαίνονται στα διαγράμματα), όπως για παράδειγμα οι Περιορισμοί (Restrictions), ορίστηκαν μέσω του Πίνακα Ιδιοτήτων. Παρακάτω παρουσιάζονται τα διαγράμματα που υλοποιήθηκαν στο πλαίσιο της οντολογίας κρασιών.



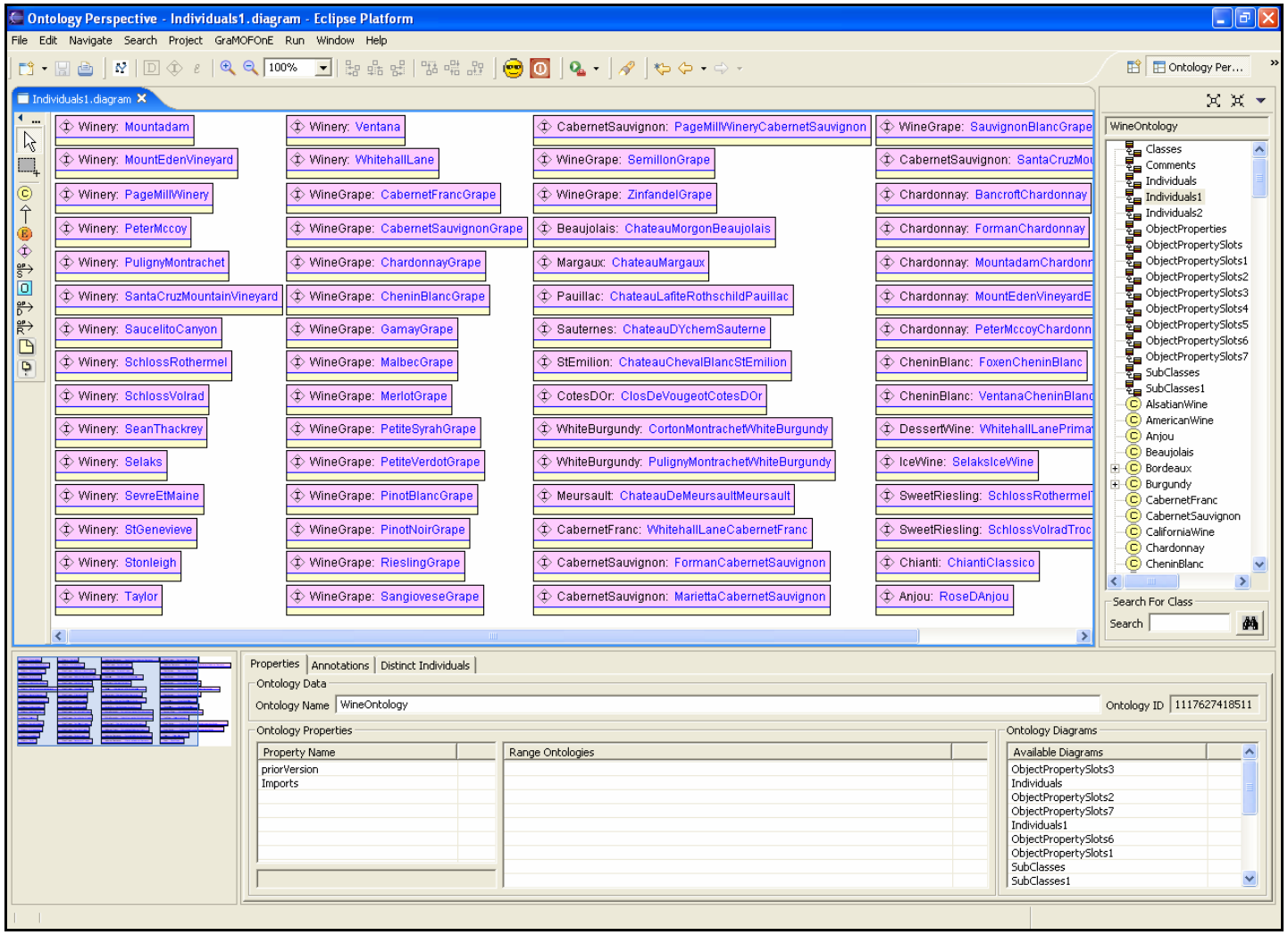

**Εικόνα 77: Διάγραμμα Κλάσεων**



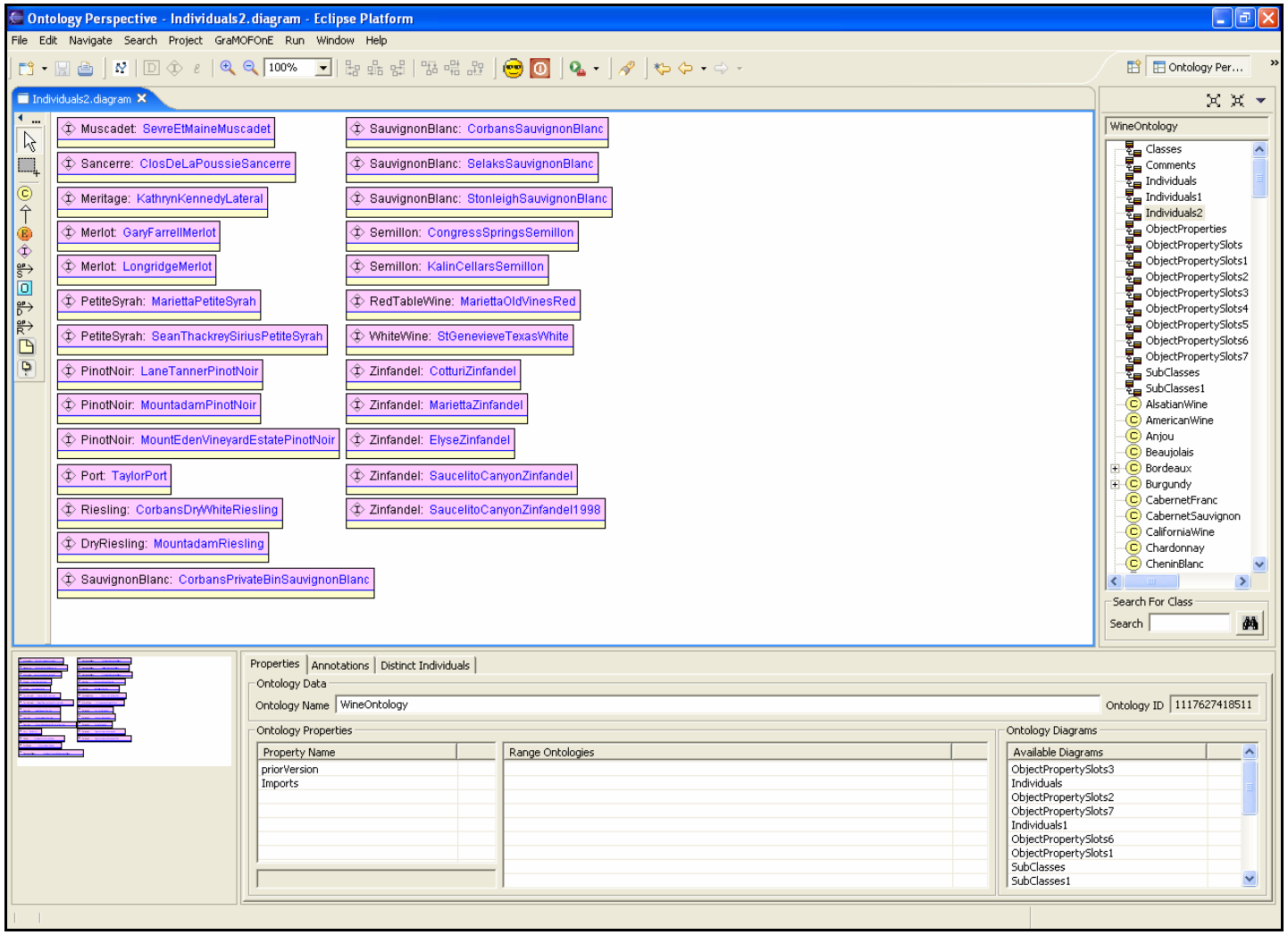

**Εικόνα 78: Διάγραμμα Σχολίων τύπου "Comment"**



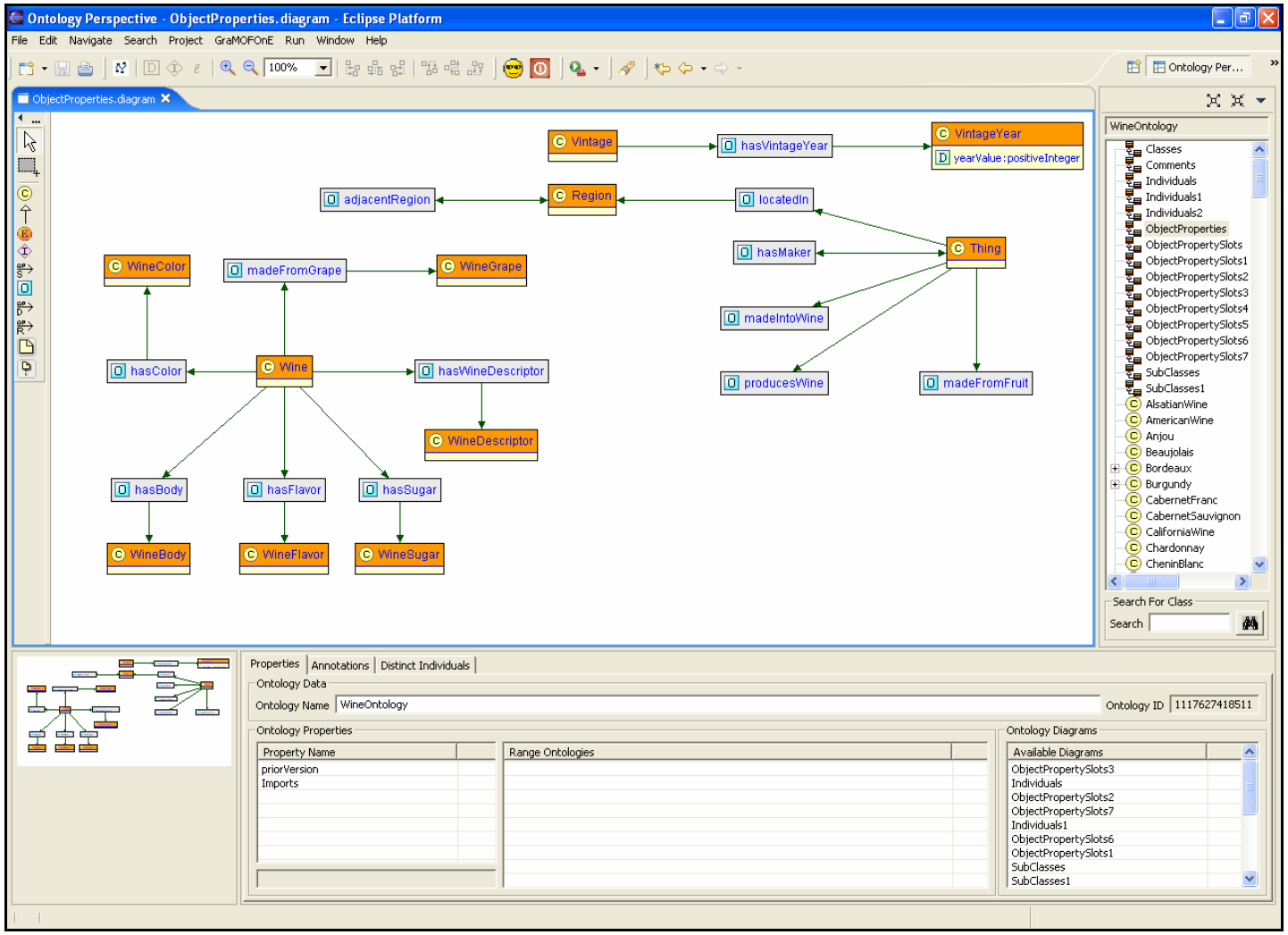

**Εικόνα 79: Διάγραμμα Στιγμιότυπων Κλάσεων (α)**



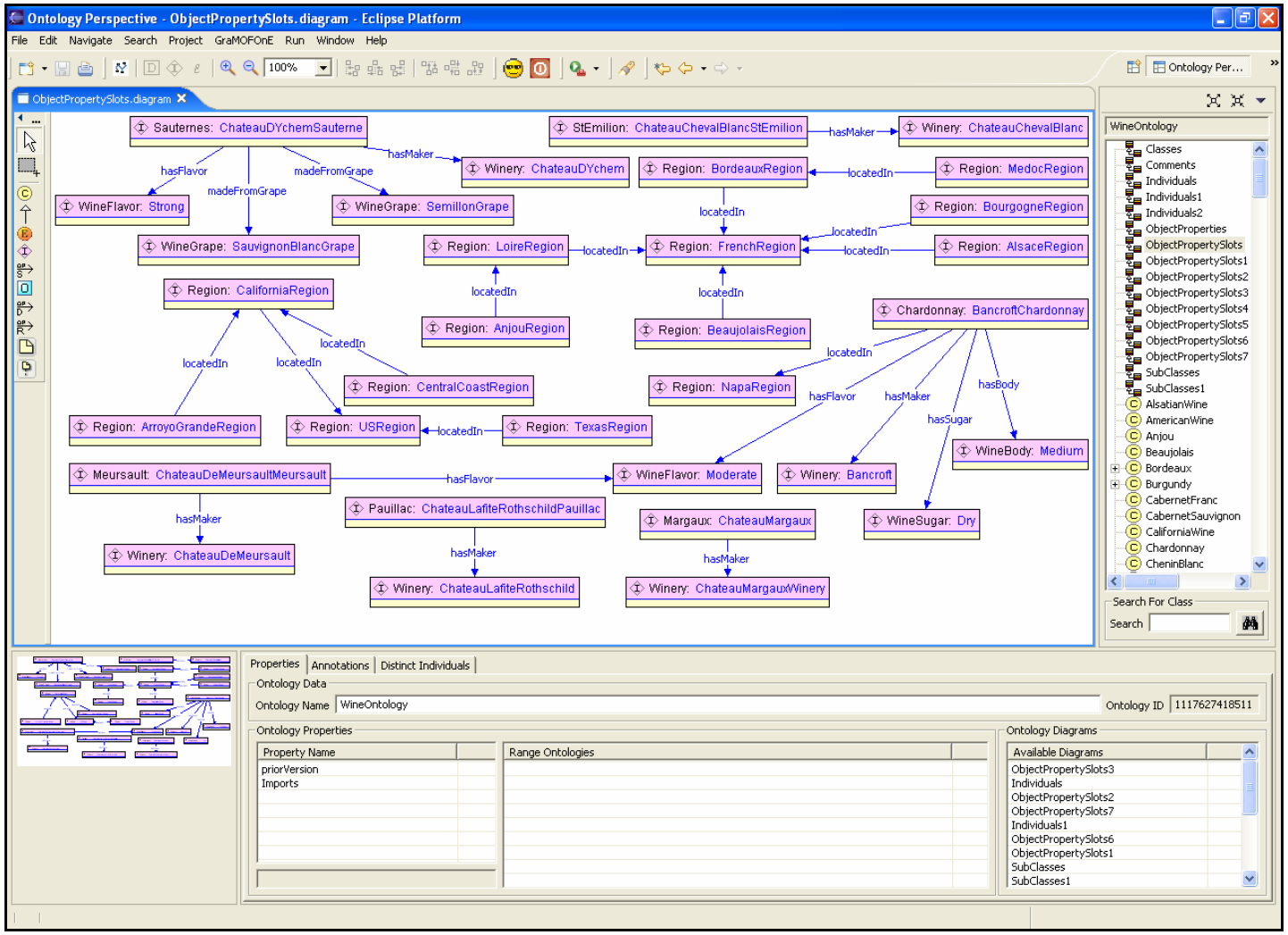

**Εικόνα 80: Διάγραμμα Στιγμιότυπων Κλάσεων (β)**



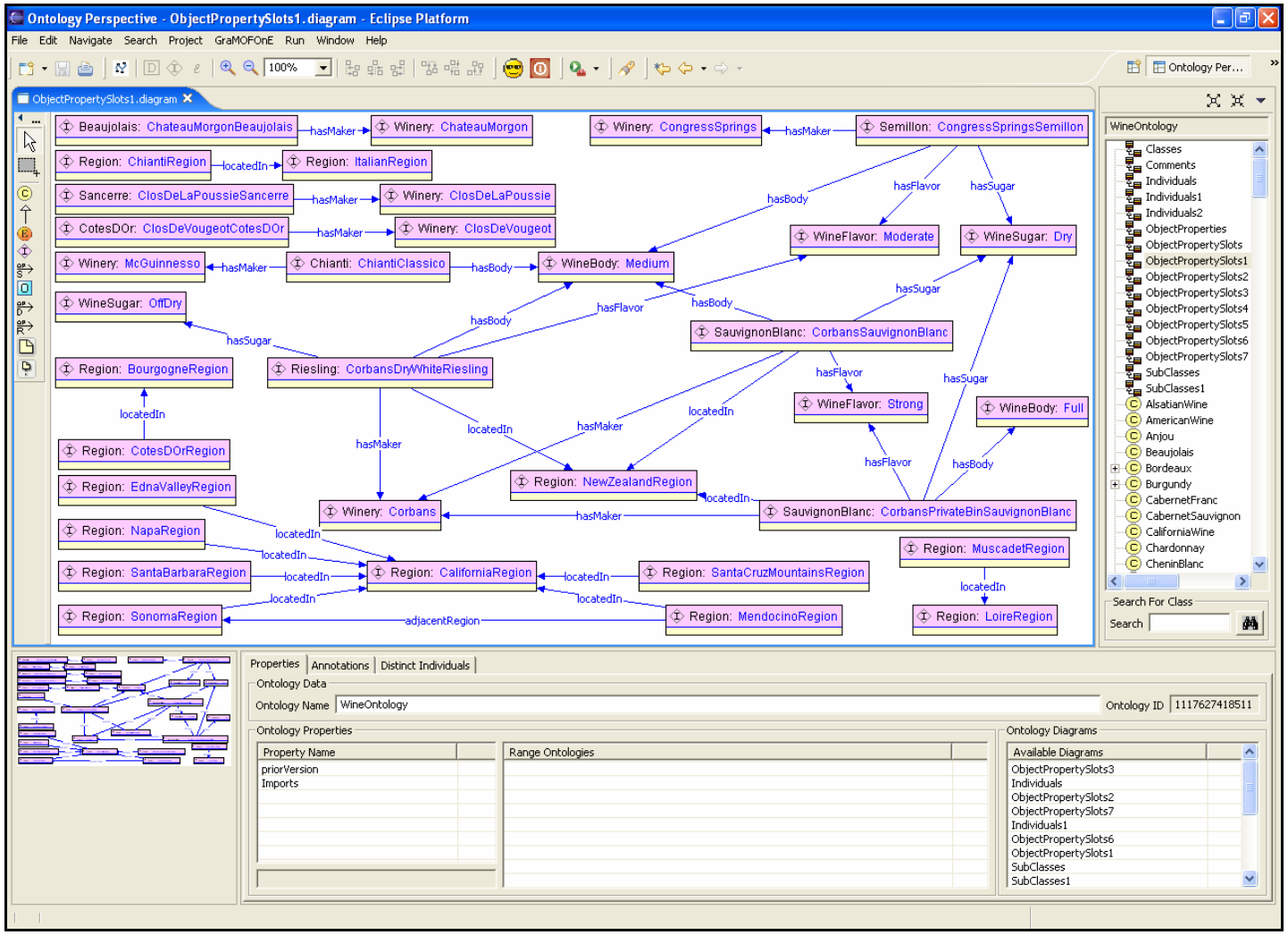

**Εικόνα 81: Διάγραμμα Στιγμιότυπων Κλάσεων (γ)**



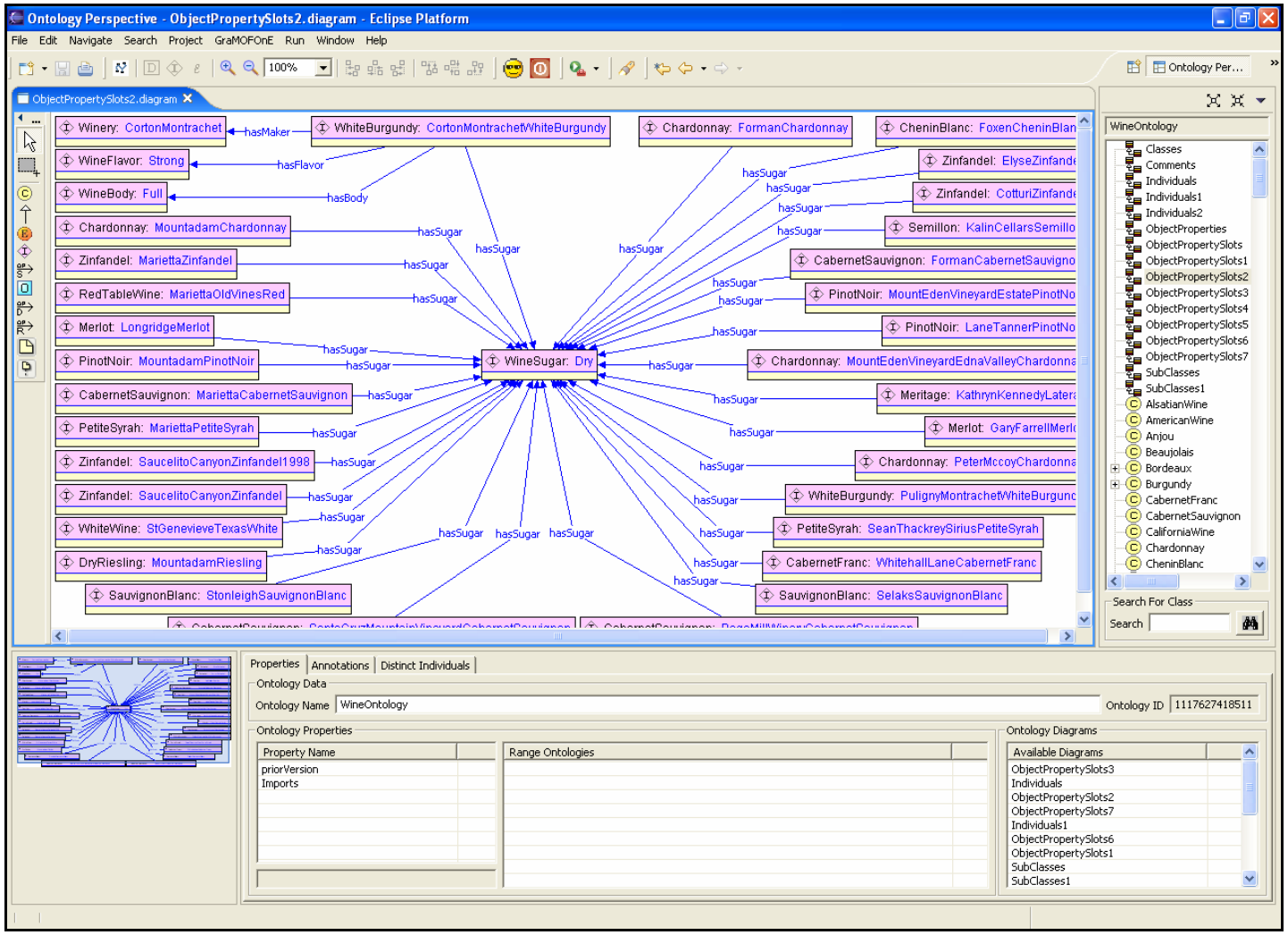

**Εικόνα 82: Διάγραμμα Ιδιοτήτων τύπου "Object Property"**



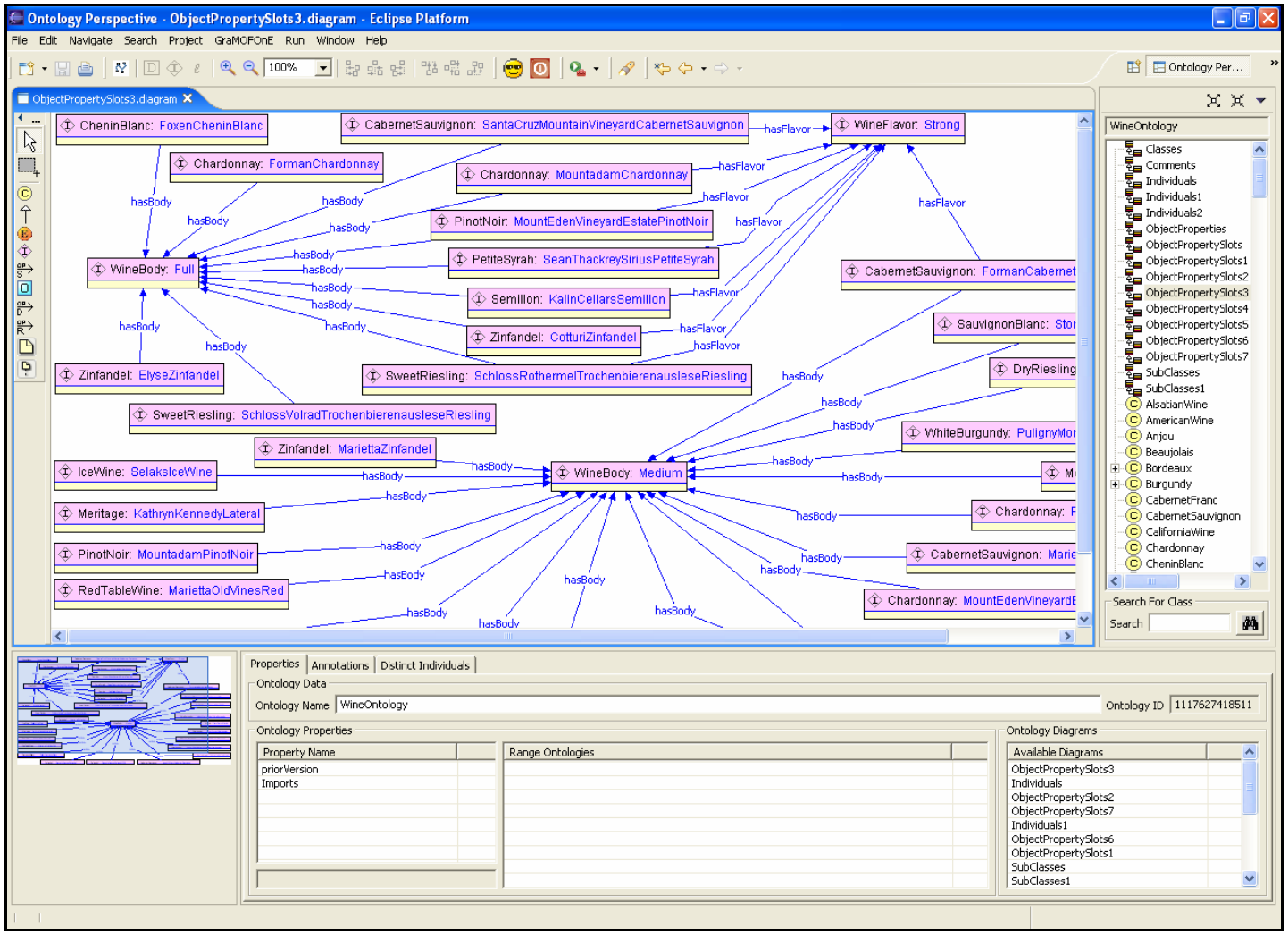

**Εικόνα 83: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (α)**



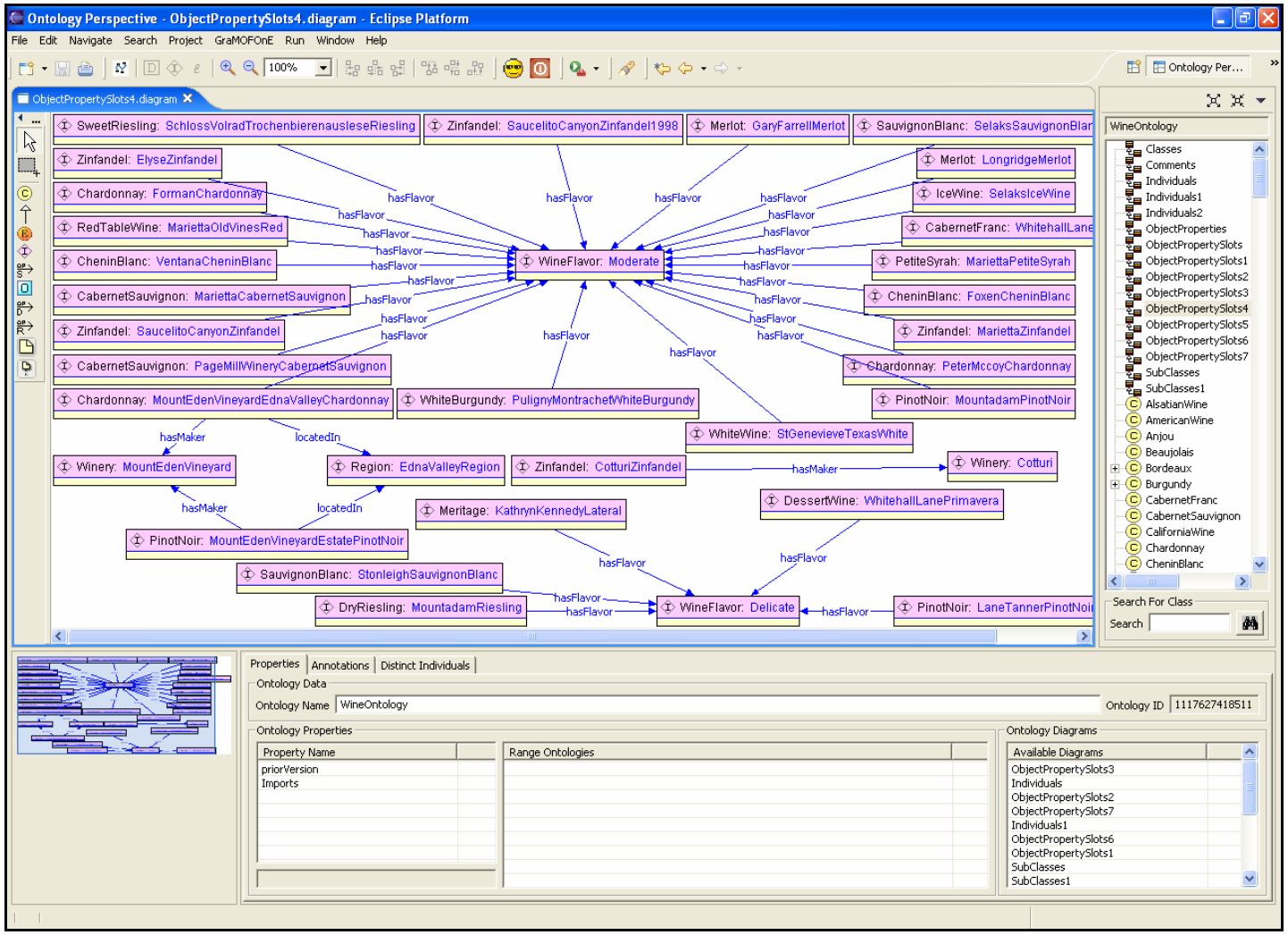

**Εικόνα 84: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (β)**



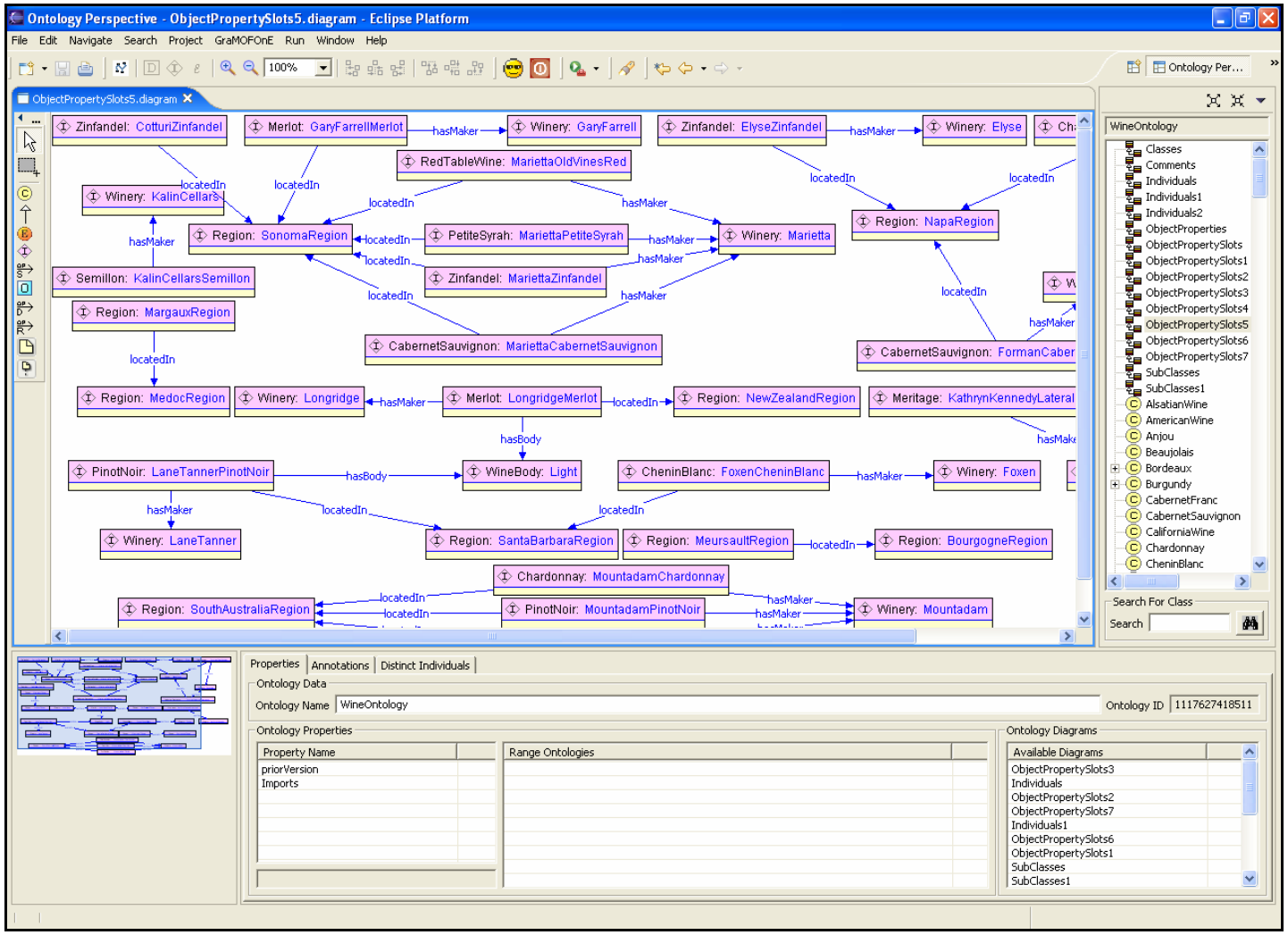

**Εικόνα 85: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (γ)**



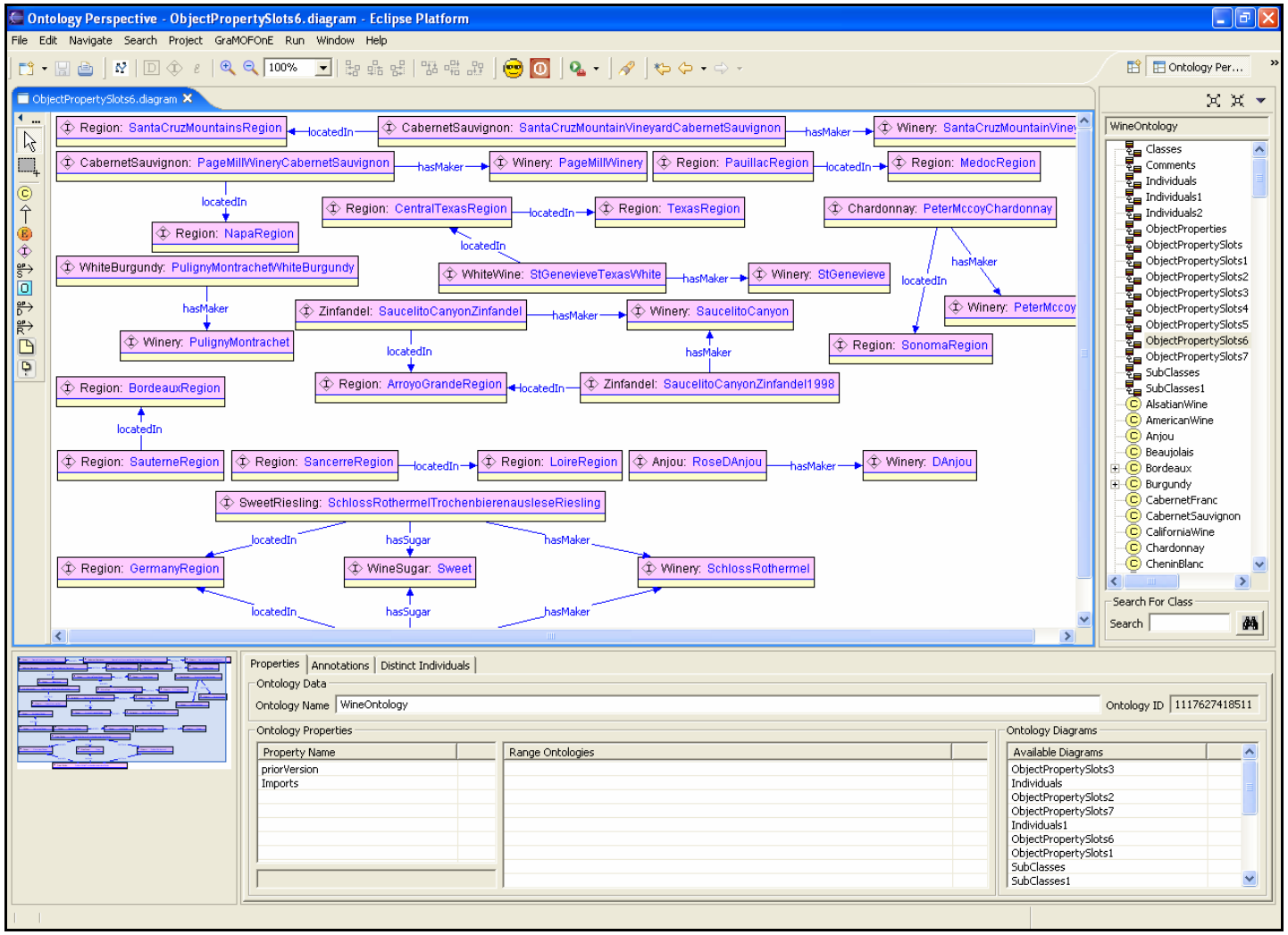

**Εικόνα 86: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (δ)**



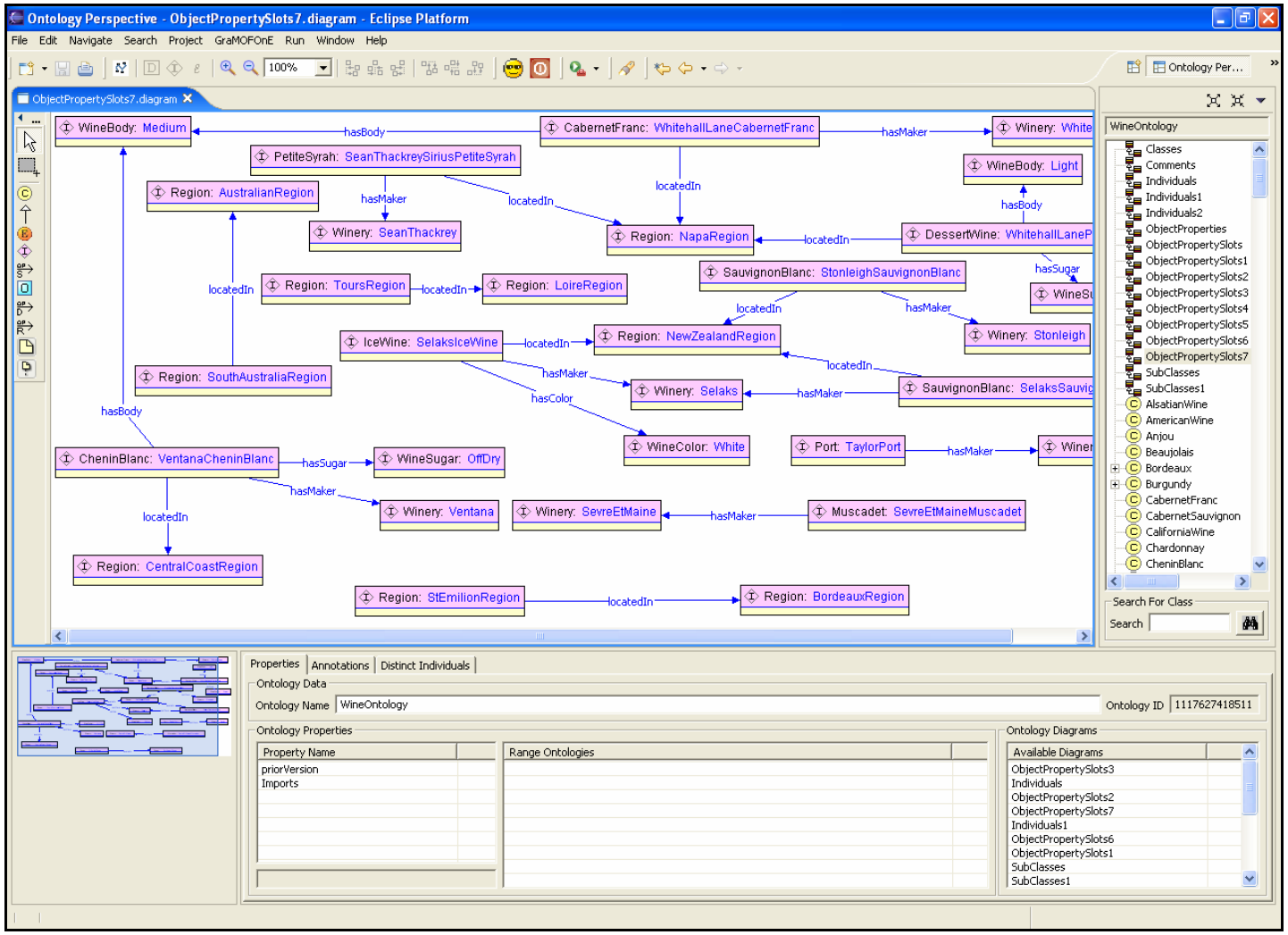

**Εικόνα 87: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (ε)**



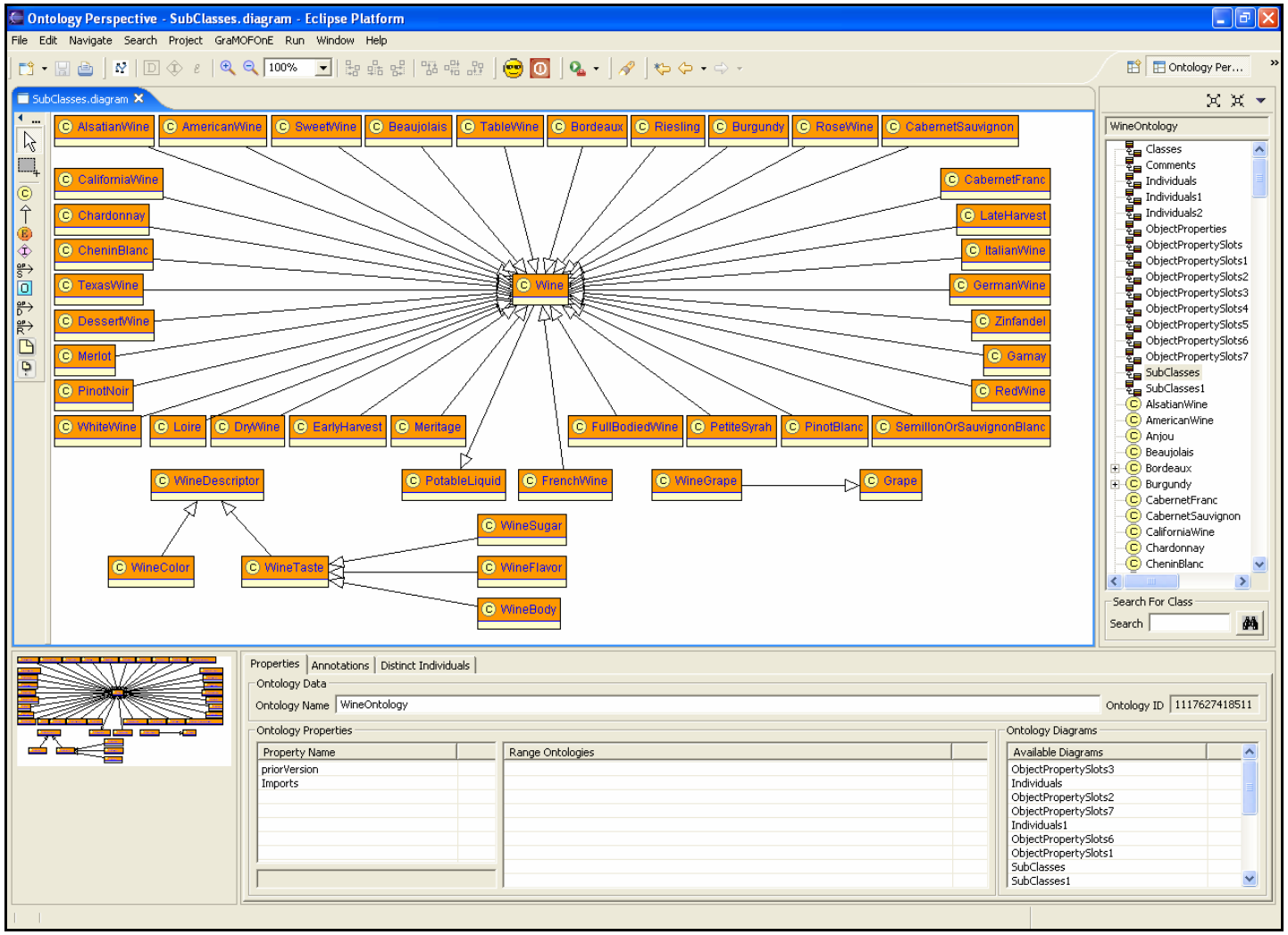

**Εικόνα 88: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (ζ)**



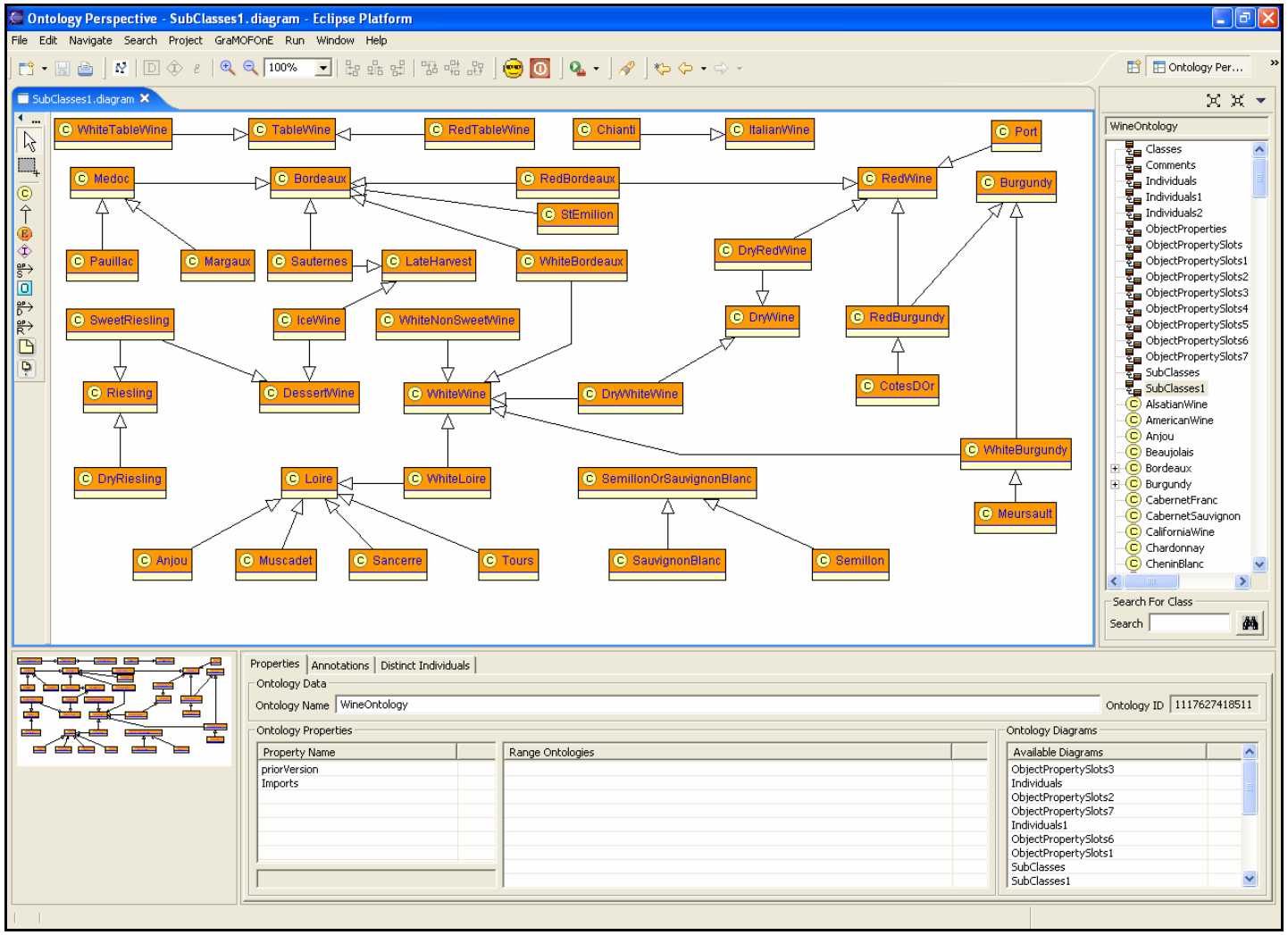

**Εικόνα 89: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (η)**




**Εικόνα 90: Διάγραμμα Στιγμιότυπων Ιδιοτήτων τύπου "Object Property" (θ)**




**Εικόνα 91: Διάγραμμα Υπό-Κλάσεων (α)**




**Εικόνα 92: Διάγραμμα Υπό-Κλάσεων (β)**

Η οντολογία ύστερα από την πετυχημένη δημιουργία της αποθηκεύτηκε στη Βάση Γνώσης (Knowledge Base) του DBE με πλήρη επιτυχία. Το XMI 1.2 έγγραφο που περιέχει την οντολογία που υλοποιήθηκε με τη χρήση του GRAMOFONE παρουσιάζεται στο Παράρτημα Β. Ύστερα από προσεκτική εξέταση και σύγκριση του RDF πρωτότυπου εγγράφου που περιέχει την οντολογία με αυτό που προκύπτει μετά την υλοποίηση της οντολογίας στο GRAMOFONE παρατηρούμε πως οι οντολογίες που περιέχονται στα δύο έγγραφα ταυτίζονται.

Η οντολογία τέλος ανακτήθηκε από τη Βάση Γνώσης με την ίδια επίσης επιτυχία.

**Αξιολόγηση του Ontology Definition Metamodel**

Στην διάρκεια υλοποίησης και χρήσης του GRAMOFONE διαπιστώθηκαν μερικά λάθη στο μετά-μοντέλο ODM τα οποία το έκαναν μη συμβατό με τη γλώσσα OWL-DL. Αν και τα λάθη αυτά δεν ήταν σημαντικά λάθη θεώρησης, ωστόσο ήταν περιοριστικά για τη χρήση του μετά-μοντέλου και εμπόδιζαν την ανάπτυξη οντολογιών συμβατών με τη γλώσσα OWL-DL. Τα λάθη που παρατηρήθηκαν είναι:

- Στο ODM ορίζεται πως η έννοια «OntologyProperty» (δηλαδή η έννοια της ιδιότητας της οντολογίας) πρέπει να έχει τουλάχιστον μία (1…*) οντολογία ορισμένη ως «Domain». Παρατηρήθηκε όμως πως αυτή η θεώρηση είναι λανθασμένη και πως η έννοια «OntologyProperty» θα πρέπει να έχει ακριβώς μία (1) οντολογία ορισμένη ως «Domain».
- Στο ODM ορίζεται πως η έννοια «Property» πρέπει να έχει ακριβώς ένα (1) «sub-Property», το οποίο είναι λάθος. Η έννοια «Property» μπορεί να έχει από κανένα έως πολλά (0…*) «sub-Properties».
- Στο ODM ορίζεται πως η έννοια «Datatype Property» μπορεί να έχει από κανένα έως πολλά (0…*) «DataRanges», το οποίο είναι λάθος. Η έννοια «Datatype Property» πρέπει να έχει ακριβώς ένα (1) «DataRange».
- Στο ODM ορίζεται πως η έννοια «Thing» χρησιμοποιείται στις σχέσεις «differentFrom_As» και «sameAs_As», το οποίο είναι λάθος. Οι σχέσεις αυτές αναφέρονται αποκλειστικά στην έννοια «ClassThing» και όχι στην «Thing».
- Στο ODM ορίζεται πως μία λίστα τύπου «AllDifferent» περιλαμβάνει στοιχεία τύπου «Thing» το οποίο είναι λάθος. Μία λίστα αυτού του είδους περιλαμβάνει στοιχεία τύπου «ClassThing».
- Στο ODM ορίζεται πως η έννοια «ClassThing» πρέπει να έχει ακριβώς ένα (1) «DatatypePropertyThing» το οποίο είναι λάθος. Η έννοια «ClassThing» μπορεί να έχει από κανένα έως πολλά (0…*) «DatatypePropertyThings».

- Στο ODM ορίζεται πως η έννοια «Object Property» πρέπει να έχει ακριβώς ένα (1) «ObjectPropertyThing», το οποίο είναι λάθος. Η έννοια «Object Property» μπορεί να έχει από κανένα έως πολλά (0…*) «ObjectPropertyThings».
- Στο ODM ορίζεται πως η έννοια «ObjectPropertyThing» πρέπει να έχει τουλάχιστον ένα (1…*) «ClassThing» ορισμένο ως «Domain» το οποίο είναι λάθος. Η έννοια «ObjectPropertyThing» πρέπει να έχει ακριβώς ένα (1) «ClassThing» ορισμένο ως «Domain».
- Στο ODM ορίζεται πως η έννοια «ClassThing» πρέπει να είναι ορισμένη ως «Range» σε ακριβώς ένα (1) «ObjectPropertyThing» το οποίο είναι λάθος. Η έννοια «ClassThing» μπορεί να είναι ορισμένη σε κανένα έως πολλά (0…*) «ObjectPropertyThings».
- Στο ODM ορίζεται πως η έννοια «Datatype Property» πρέπει να έχει ακριβώς ένα (1) «DatatypePropertyThing» το οποίο είναι λάθος. Η έννοια «Datatype Property» μπορεί να έχει από κανένα έως πολλά (0…*) «DatatypePropertyThings» .
- Στο ODM ορίζεται πως η έννοια «Enumeration» περιέχει «Literals», το οποίο είναι λάθος επίσης. Η έννοια «Enumeration» πρέπει να περιέχει «Plain Literals».

Κατά τη διάρκεια ανάπτυξης της οντολογίας κάθε λάθος που διαπιστωνόταν διορθωνόταν άμεσα στο μετά-μοντέλο ODM και κατόπιν στη Βάση Γνώσης που υλοποιεί το μετά-μοντέλο. Έτσι, ήταν δυνατή η πλήρης ανάπτυξη της οντολογίας των κρασιών η οποία παρατέθηκε στην προηγούμενη ενότητα.

**Παραδοχές Υλοποίησης**

Κατά τη διάρκεια υλοποίησης του GRAMOFONE έγιναν κάποιες παραδοχές, όσον αφορά κάποιες έννοιες του ODM, οι οποίες έγιναν είτε λόγο της υλοποίησης που αποφασίστηκε για το εργαλείο είτε επειδή εννοιολογικά δεν υπήρχε λόγος να υλοποιηθούν αυτές οι έννοιες στο εργαλείο. Οι παραδοχές που έγιναν είναι οι εξής:

- Στο ODM ορίζεται πως η έννοια «Datatype Property» μπορεί να έχει από καμία έως πολλές (0…*) «Domain» ODM Κλάσεις. Η έννοια «Datatype Property» υλοποιήθηκε γραφικά, όπως περιγράφηκε ήδη, να περιέχεται στις ODM Κλάσεις οι οποίες είναι οι «Domain» Κλάσεις της. Κατά συνέπεια για να οριστεί ένα «Datatype Property» θα πρέπει πρώτα να έχει οριστεί η ODM Κλάση η οποία θα είναι «Domain» για το «Datatype Property», δηλαδή ένα «Datatype Property» δεν μπορεί να οριστεί ανεξάρτητα από την ή τις «Domain» ODM Κλάσεις του. Στο μοντέλο κατά συνέπεια του GRAMOFONE έχει οριστεί πως ένα «Datatype Property» θα πρέπει να έχει τουλάχιστον μία (1…*) ODM Κλάση σαν «Domain».

- Οι έννοια «DeprecatedDatatype» δεν έχει υλοποιηθεί καθώς δεν ορίζονται τέτοιοι τύποι δεδομένων από το ODM μιας και έχει γίνει η παραδοχή ότι χρησιμοποιούνται οι τύποι δεδομένων που ορίζονται από την γλώσσα XML Schema.
- Το χαρακτηριστικό (attribute) «Language: String» που ορίζεται από το ODM για τα «Plain Literals» τίθεται εξ' ορισμού στην τιμή "English" από το GRAMOFONE.

**Ανακεφαλαίωση**

Στο κεφάλαιο παρουσιάστηκε η διαδικασία και τα αποτελέσματα της αξιολόγησης που έγινε τόσο για το μετά-μοντέλο ODM όσο και για το εργαλείο GRAMOFONE. Η αξιολόγηση αυτή έγινε με την ανάπτυξη μιας οντολογίας με τη χρήση του εργαλείου και την αποθήκευση της στη Βάση Γνώσης. Σκοπός της αξιολόγησης αυτής ήταν να εξεταστεί η πληρότητα και η ορθότητα του μετά-μοντέλου καθώς και η συμβατότητά του με τη γλώσσα OWL. Όσον αφορά το εργαλείο GRAMOFONE, στόχος της αξιολόγησης ήταν να ελεγχθεί η πληρότητά του, η αξιοπιστία του, και η χρηστικότητά του. Η οντολογία που αναπτύχθηκε είναι μια οντολογία κρασιών (wine ontology) η οποία είναι μία ευρέως χρησιμοποιημένη οντολογία η οποία δίδεται από το πανεπιστήμιο Stanford ως οντολογία δοκιμών (testing ontology) για εργαλεία σχετικά με οντολογίες (ontology editors, reasoners, etc.). Επιπλέον παρουσιάστηκαν οι παραδοχές που έγιναν στο πλαίσιο του GRAMOFONE οι οποίες θέτουν κάποιες μικρές διαφοροποιήσεις του μοντέλου της εφαρμογής από αυτό που ορίζεται στο ODM.

Κεφάλαιο 9

# ΑΝΑΚΕΦΑΛΑΙΩΣΗ – ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

**Συμπεράσματα – Ανακεφαλαίωση**

Στο πλαίσιο της παρούσης διπλωματικής εργασίας αναπτύχθηκε ένα γραφικό εργαλείο ανάπτυξης και επεξεργασίας οντολογιών (πάνω στην πλατφόρμα Eclipse) με βάση το μετά-μοντέλο ODM (το οποίο είναι βασισμένο στην αρχιτεκτονική MOF) και με χρήση σημειολογίας παρόμοιας με αυτή που προτείνεται από τη γλώσσα μοντελοποίησης UML και υποστηρίζεται ευρέως από πολλά εμπορικά εργαλεία (π.χ. Rational Rose, UML Magic Draw, Poseidon κ.α.). Οι οντολογίες που δημιουργούνται ή τροποποιούνται με το εργαλείο αυτό θα πρέπει να αποθηκεύονται στη βάση δεδομένων του DBE που υποστηρίζει το μετά-μοντέλο ODM. Οι βασικοί στόχοι της διπλωματικής ήταν οι εξής:

- Να μελετήσει το μετά-μοντέλο αναπαράστασης οντολογιών ODM, και να προσδιορίσει τις απαιτήσεις που πρέπει να ικανοποιηθούν από ένα εργαλείο ανάπτυξης και επεξεργασίας οντολογιών με βάση το μετά-μοντέλο αυτό.
- Να σχεδιάσει και να υλοποιήσει ένα εργαλείο το οποίο θα επιτρέπει στο χρήστη τη δημιουργία οντολογιών (οι οποίες θα ακολουθούν το ODM) με γραφικό τρόπο, και συγκεκριμένα με χρήση σημειολογίας παρόμοιας με αυτή που προτείνεται από τη γλώσσα UML.
- Να παρέχει τη δυνατότητα αποθήκευσης και ανάκτησης των οντολογιών που δημιουργούνται από το γραφικό εργαλείο σε μια Βάση Γνώσης η οποία είναι βασισμένη στο πρότυπο MOF και υλοποιεί το μετά-μοντέλο ODM παρέχοντας ένα σύνολο διεπαφών σύμφωνα με το πρότυπο JMI (Java Metadata Interface).
- Να αξιολογήσει την πληρότητα και την ορθότητα του μετά-μοντέλου ODM, καθώς και τη συμβατότητά του με τη γλώσσα OWL-DL.

Για την ομαλή εισαγωγή του αναγνώστη στο αντικείμενο που πραγματεύεται η συγκεκριμένη διπλωματική εργασία έγινε αρχικά μια παρουσίαση του γενικού πλαισίου εργασίας όπου και παρουσιάστηκαν έννοιες και τεχνολογίες σχετικές με την παρούσα διπλωματική. Συγκεκριμένα έγινε μια γρήγορη παρουσίαση της έννοιας των οντολογιών καθώς και των

σχετικών με αυτή τεχνολογιών (RDF, OWL, κ.α.). Επίσης, παρουσιάστηκαν έννοιες οι οποίες αφορούν σε γενικότερες αρχές μοντελοποίησης και οργάνωσης μετά-δεδομένων όπως η αρχιτεκτονική μεταδεδομένων του MOF, η γλώσσα μοντελοποίησης UML, κ.α. προκειμένου να εξοικειωθεί ο αναγνώστης με τις τεχνολογίες αυτές αλλά και να κατανοήσει το πως αυτές σχετίζονται με το μετά-μοντέλο ODM το οποίο και υλοποιείται από το εργαλείο που αναπτύχθηκε στο πλαίσιο της διπλωματικής αυτής εργασίας. Η παρουσίαση του πλαισίου εργασίας ολοκληρώθηκε με την παρουσίαση του ίδιου του μετά-μοντέλου ODM. Η παρουσίαση έγινε με αρκετά λεπτομερή τρόπο δεδομένου ότι πρόκειται για το βασικό μοντέλο που καθορίζει τη λειτουργικότητα που πρέπει να υποστηρίζεται από το εργαλείο που αναπτύχθηκε.

Στη συνέχεια παρατέθηκε η επισκόπηση της σχετικής με την παρούσα διπλωματική έρευνας. Συγκεκριμένα έγινε μια προσπάθεια παράθεσης των σχετικών εργασιών (εργαλείων) και αξιολόγησης των πλεονεκτημάτων και μειονεκτημάτων της καθεμιάς. Κατόπιν, και εφόσον είχαν παρουσιαστεί οι στόχοι της διπλωματικής εργασίας, το πλαίσιο εργασίας της, και το μετά-μοντέλο ODM έγινε η ανάλυση απαιτήσεων με την καταγραφή των διαφόρων ενδιαφερόμενων ρόλων (stakeholders) και φυσικά των λειτουργιών που πρέπει να υποστηρίζονται από το εργαλείο. Η περιγραφή των λειτουργιών έγινε με τη χρήση της τεχνολογίας των χρησίτυπων (Use Cases) η οποία είναι η πλέον διαδεδομένη τεχνολογία ανάλυσης απαιτήσεων λογισμικού.

Ακολούθησε η εκτενής περιγραφή της αρχιτεκτονικής του εργαλείου καθώς και του γραφικού περιβάλλοντος εργασίας που αυτό προσφέρει. Το εργαλείο υλοποιήθηκε ως ένα επιπρόσθετο (plug-in) της πλατφόρμας Eclipse. Η ακριβής αρχιτεκτονική του εργαλείου και η σχέση του με την πλατφόρμα Eclipse καθώς και άλλα επιπρόσθετα (όπως το GEF για παράδειγμα που προσφέρει βασικές λειτουργίες διαχείρισης γραφικών) παρουσιάστηκε και έγινε εκτενής περιγραφή των διαφόρων δομικών της υπό-μονάδων. Επίσης, στη συνέχεια παρουσιάστηκε αναλυτικά το γραφικό περιβάλλον εργασίας (Graphical User Interface) που παρέχεται από το εργαλείο. Η παρουσίαση έγινε με τη χρήση στιγμιότυπων (screenshots) τα οποία καταδεικνύουν το πως διεκπεραιώνονται οι διάφορες λειτουργίες που υποστηρίζονται από το εργαλείο (σύμφωνα πάντα με την ανάλυση απαιτήσεων που έγινε προηγούμενα).

Τέλος παρουσιάστηκε η αξιολόγηση τόσο του μετά-μοντέλου ODM όσο και του εργαλείου που επιχειρήθηκε στο πλαίσιο της παρούσας διπλωματικής εργασίας. Η αξιολόγηση έγινε με την ανάπτυξη μιας πρότυπης οντολογίας κρασιών η οποία παρέχεται από το πανεπιστήμιο Stanford και η οποία κάνει χρήση των περισσοτέρων δυνατοτήτων της γλώσσας OWL-DL. Η οντολογία αυτή είναι ευρέως αποδεκτή για τη δοκιμή εργαλείων σχετικών με οντολογίες.

Η διαδικασία της αξιολόγησης ανέδειξε μερικά προβλήματα του μετά-μοντέλου ODM τα οποία και παρουσιάστηκαν στη σχετική ενότητα. Επίσης, η αξιολόγηση αυτή ανέδειξε και διάφορα προβλήματα του εργαλείου, τα οποία και διορθώθηκαν.

**Μελλοντικές Επεκτάσεις**

Το εργαλείο που υλοποιήθηκε στο πλαίσιο της παρούσας διπλωματικής εργασίας δίνει τη δυνατότητα δημιουργίας και επεξεργασίας οντολογιών με τρόπο γραφικό και εύκολα κατανοητό στο χρήστη. Το εργαλείο καλύπτει σχεδόν πλήρως το μετά-μοντέλο ODM και την εκφραστικότητα που αυτό προσφέρει. Ωστόσο υπάρχουν μερικές λειτουργίες οι οποίες είτε δεν υποστηρίζονται πλήρως, είτε θα μπορούσαν να υποστηριχθούν καλύτερα και αποτελούν ενδιαφέροντες μελλοντικές επεκτάσεις του εργαλείου. Συγκεκριμένα οι επόμενες επεκτάσεις θεωρούνται αρκετά ενδιαφέρουσες:

- Εισαγωγή οντολογιών που είναι περιγεγραμμένες με τη γλώσσα OWL-DL στο εργαλείο. Το εργαλείο στην παρούσα έκδοσή του εισάγει οντολογίες που είναι περιγεγραμμένες με τη γλώσσα ODM. Το ODM αποτελεί μία γλώσσα συμβατή με την OWL-DL. Η συμβατότητα αυτή είναι επιθυμητή καθώς με ορισμένες μικρές μεταβολές στο εργαλείο GRAMOFONE θα είναι δυνατή η εισαγωγή οντολογιών περιγεγραμμένων με τη γλώσσα OWL-DL. Η λειτουργία αυτή κρίνεται ιδιαίτερα σημαντική με γνώμονα την αξία και τη διαρκώς αυξανόμενη χρήση της γλώσσας OWL-DL σήμερα.
- Εξαγωγή οντολογιών που δημιουργούνται με το εργαλείο σε σύνταξη OWL-DL. Σύμφωνα με την προηγούμενη προτεινόμενη μελλοντική επέκταση είναι προφανής η επιδίωξη όχι μόνο της εισαγωγής αλλά και της εξαγωγής οντολογιών που είναι περιγεγραμμένες με τη γλώσσα OWL-DL για τους ίδιους ακριβώς λόγους που περιγράφηκαν πιο πάνω.
- Καλύτερη υποστήριξη της δημιουργίας Individuals. Να υπάρχει μεγαλύτερη συνέπεια μεταξύ των χαρακτηριστικών που ορίζονται για τα Individuals και των ορισμών των κλάσεων ή των properties που προσδιορίζουν το περιεχόμενο του κάθε Individual. Θα μπορούσαν να υλοποιηθούν κατάλληλοι μηχανισμοί (wizards) για το σκοπό αυτό.
- Παροχή μηχανισμού ελέγχου συνέπειας της οντολογίας. Το εργαλείο GRAMOFONE ενισχύει τη λειτουργικότητά του με μηχανισμούς που αποσκοπούν στην αποτροπή του χρήστη από το να δημιουργήσει ασυνέπειες στην οντολογία που δημιουργεί. Οι μηχανισμοί αυτοί λειτουργούν πάνω στις επιλογές του χρήστη. Ωστόσο πιθανώς να υπάρχουν τρόποι να δημιουργηθούν ασυνέπειες στη διάρκεια

δημιουργίας μίας οντολογίας παρόλη την ύπαρξη των μηχανισμών αυτών. Για το λόγο αυτό, θα ήταν πολύ χρήσιμος ένας μηχανισμός που θα ελέγχει τη συνέπεια της οντολογίας στην πληρότητά της (σαν σύνολο).

- Δυνατότητα εμφάνισης μέσα στο εργαλείο των οντολογιών που χρησιμοποιούνται (import) από την υπό-επεξεργασία οντολογία. Στην παρούσα έκδοση του GRAMOFONE όλοι οι όροι μίας οντολογίας, που υλοποιείται από το χρήστη, ανήκουν στην οντολογία αυτή, δηλαδή κάθε όρος της οντολογίας έχει δημιουργηθεί από το μηδέν. Δεν έχει προβλεφθεί η χρήση (εισαγωγή) όρων οι οποίοι να ανήκουν σε άλλες οντολογίες. Θα ήταν κατά συνέπεια πολύ χρήσιμη η δυνατότητα εισαγωγής (import) οντολογιών, στο πλαίσιο δημιουργίας μίας νέας οντολογίας, και η δυνατότητα χρησιμοποίησης των όρων της εισαγόμενης οντολογίας.
- Δυνατότητα αυτόματης αναπαράστασης διαγραμμάτων. Θα ήταν χρήσιμη η υλοποίηση ενός μηχανισμού για την αυτόματη δημιουργία διαγραμμάτων τα οποία θα αναπαριστούν γραφικά οντολογίες οι οποίες όταν ανακτώνται με το εργαλείο δεν θα περιλαμβάνουν διαγράμματα για τη γραφική τους αναπαράσταση.
- Δυνατότητα επικοινωνίας με τη Βάση Γνώσης σε μικρότερο «granularity» από εκείνο της οντολογίας, δηλαδή να είναι δυνατή η ανάκτηση ενός τμήματος της οντολογίας (Κλάσεων, Ιδιοτήτων (Properties) κλπ. ή ακόμα και ενός διαγράμματος) και όχι υποχρεωτικά ολόκληρης της οντολογίας.
- Υποστήριξη ταυτόχρονης και παράλληλης προβολής ή επεξεργασίας του περιεχομένου μίας οντολογίας από πολλούς χρήστες (Concurrency).
- Δυνατότητα πρόσβασης στο εργαλείο από απόσταση (Remote Access).
- Παροχή συστήματος βοήθειας (help system). Παρόλο που το γραφικό περιβάλλον εργασίας του GRAMOFONE είναι ιδιαίτερα φιλικό προς το χρήστη, έτσι ώστε ο χρήστης να είναι σε θέση να προσαρμόζεται γρήγορα και εύκολα στο περιβάλλον εργασίας του, είναι επιθυμητή η ύπαρξη ενός συστήματος βοήθειας το οποίο θα παρέχει πληροφορίες για το εργαλείο και τις λειτουργίες του με παραδείγματα και οδηγούς για την εκτέλεση των λειτουργιών.

# ΒΙΒΛΙΟΓΡΑΦΙΑ


[1] http://www.digital-ecosystem.org/html/ (The Research Project)

[2] TUC/MUSIC, "DBE Knowledge Representation Models", DBE Project Deliverable: https://dbe.digital-ecosystem.net/files/documents/8/570/file_570.dat?filename=Del_14.1_DBE_DBE_Knowledge_Representation_Models.pdf

[3] Michael Denny: *Ontology Tools Survey, Revisited*, July 14, 2004, http://www.xml.com/pub/a/2004/07/14/onto.html

[4] A. J. Duineveld, R. Stoter, M. R. Weiden, B. Kenepa and V. R. Benjamins: *WonderTools? A comparative study of ontological engineering tools,* Dept. of Social Science Informatics, University of Amsterdam, The Netherlands

[5] Protégé Official Home Page, http://protege.stanford.edu/overview/index.html

[6] Protégé Owl Plug-in Official Home Page, http://protege.stanford.edu/plugins/owl/

[7] Protégé ezOWL Plug-in Official Home Page, http://iweb.etri.re.kr/ezowl/#Introduction

[8] Construct Tool Official Home Page, http://www.networkinference.com/products/construct_it.html

[9] IsaViz Tool Official Home Page, http://www.w3.org/2001/11/IsaViz/

[10] OI-modeler Tool Official Home Page, http://kaon.semanticweb.org/users

[11] MR3 Tool Official Home Page, http://panda.cs.inf.shizuoka.ac.jp/mmm/mr3/index.html

[12] OntoTrack Tool Official Home Page, http://www.informatik.uni-ulm.de/ki/ontotrack/

[13] RDFAuthor Tool Official Home Page, http://rdfweb.org/people/damian/RDFAuthor/

[14] OWL Web Ontology Language Overview, W3C Recommendation 10 February 2004, Deborah L. McGuinness (Knowledge Systems Laboratory, Stanford University), Frank van Harmelen (Vrije Universiteit, Amsterdam)

[15] Martin Fowler, Kendall Scott: ΕΙΣΑΓΩΓΗ ΣΤΗ UML, ΔΕΥΤΕΡΗ ΑΜΕΡΙΚΑΝΙΚΗ ΕΚΔΟΣΗ, ΣΥΝΟΠΤΙΚΟΣ ΟΔΗΓΟΣ ΤΗΣ ΠΡΟΤΥΠΗΣ ΓΛΩΣΣΑΣ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ ΑΝΤΙΚΕΙΜΕΝΩΝ, εκδόσεις "ΚΛΕΙΔΑΡΙΘΜΟΣ"

[16] TUC/MUSIC, "First Prototype Implementation of the DBE Knowledge Base", https://dbe.digital-ecosystem.net/files/documents/8/572/file_572.dat?filename=Del_14.2_DBE_First_Prototype_Implementation_of_the_DBE_Knowledge_Base.pdf

[17] Eclipse Platform Technical Overview, Object Technology International Inc., February 2003 (updated for 2.1; originally published July 2001).

[18] Randy Hudson, Software developer, IBM: *Create an Eclipse-based application using the Graphical Editing Framework*, 29 July 2003.

[19] GEF Overview, http://www.eclipse.org/gef/

[20] Deborah J. Mayhew: *Principles and Guidelines in Software User Interface Design*, Prentice-Hall Inc., 1991.

[21] Alistair Cockburn: *Writing Effective Use Cases.*

[22] Tom Gruber: *What is an Ontology*, http://www-ksl.stanford.edu/kst/what-is-an-ontology.html#1

[23] Renato Iannella: *An Idiot's Guide to the Resource Description Framework*, 25 January 1999.

[24] Meta Object Facility (MOF) Specification, Version 1.4, April 2002.

[25] Μέθοδος Booch: http://www.ifra.ing.tu-bs.de/docs/BoochReferenz/

[26] Μέθοδος Rumbaugh: http://www.iconixsw.com/Rumbaugh.html

[27] Μέθοδος Jacobson: http://www.iconixsw.com/Jacobson.html

[28] George Anestis, Themis Dakanalis, Nektarios Gioldasis, Nikos Pappas, Fotis Kazasis: *WP15: Business Modeling Language BML Editor*, 3 December 2004.

[29] Phil Zoio: *Building a Database Schema Diagram Editor with GEF*, September 27, 2004.

[30] UML Resource Page, http://www.uml.org/