ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Σημασιολογική αναζήτηση σε Βάση Δεδομένων με χρήση Οντολογικής γνώσης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Παντελής Σ. Κουτρουμπής

Επιβλέπον : Στέφανος Δ. Κόλλιας
Καθηγητής ΕΜΠ

Αθήνα Ιούνιος 2005

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Σημασιολογική αναζήτηση σε Βάση Δεδομένων με χρήση Οντολογικής γνώσης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Παντελής Σ. Κουτρουμπής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 13η Ιουνίου 2005

.....	.....	.....
Σ.Κόλλιας	Α.Σταφυλοπάτης	Π.Τσανάκας
Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ

Αθήνα Ιούνιος 2005

.....

Παντελής Σ. Κουτρουμπής
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών ΕΜΠ

Copyright © Παντελής Σ. Κουτρουμπής, 2005
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν το συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσοβίου Πολυτεχνείου.

ΠΕΡΙΛΗΨΗ

Το πόνημα τούτο επικεντρώνεται στην προσπάθεια προσθήκης σημασιολογικής πληροφορίας σε συστήματα βάσεων δεδομένων που δεν την διαθέτουν εξ αρχής. Όπως είναι γνωστό ένα σύστημα βάσεως δεδομένων είναι σε θέση να διαχειρίζεται με μια πληθώρα παρεχόμενων διαδικασιών τα δεδομένα που αποθηκεύονται σε αυτό χωρίς ωστόσο να δίνει την απαραίτητη σημασιολογία στα δεδομένα καθαυτά. Η μετάβαση από την αποκλειστικά ανθρωποκεντρική θεώρηση των δεδομένων σε μια ταυτόχρονα μηχανοκεντρική και ανθρωποκεντρική προσέγγιση επιτυγχάνεται με την χρήση των Οντολογιών.

Στα πλαίσια της δημιουργίας ενός σημασιολογικά επεξεργάσιμου τμήματος πληροφοριών, απαιτείται η προσθήκη ή ο υπομνηματισμός των δεδομένων με συγκεκριμένη πληροφορία. Τούτο μπορεί να επιτευχθεί στην περίπτωση των δικτυακών τόπων και των ιστοσελίδων με την χρήση της τεχνολογίας RDF (Resource Description Framework) και στην περίπτωση των βάσεων γνώσης και δεδομένων σε τοπικό ή δικτυακό επίπεδο με χρήση της OWL (Ontology Web Language). Ο τρόπος και οι διαδικασίες της προσθήκης της επιπλέον πληροφορίας είναι το μείζον θέμα της εργασίας αυτής.

Χρησιμοποιήθηκε σαν παράδειγμα χρήσης μια βάση δεδομένων από Oracle Schema σε ένα συγκεκριμένο πεδίο ενδιαφέροντος. Ιδιαίτερη προσοχή δόθηκε στο ότι το συγκεκριμένο παράδειγμα περιείχε όλους τους πιθανούς περιορισμούς που μπορεί να βρεθούν σε ένα σχήμα βάσης δεδομένων. Η διαδικασία περιλάμβανε το migration σε ένα εργαλείο επεξεργασίας και ανάπτυξης Οντολογιών. Το εργαλείο που προτιμήθηκε ήταν το Protégé 3.0. Για την διαδικασία μετάβασης χρησιμοποιήθηκε ειδικό πρόγραμμα που να δύναται να μεταφέρει πληροφορία από την βάση δεδομένων Oracle 9.2 σε περιβάλλον Protégé χωρίς να υπάρχει καμία απώλεια πληροφορίας.

Η αρχικά εισηγμένη δομή του εν λόγω σχήματος δεν ήταν δυνατό να πληροί τις απαιτήσεις μια σημασιολογικά χρήσιμης οντολογίας, καθώς η βάση αυτούσια δεν περιέχει πουθενά καμία περιγραφή των τρόπων χρήσης των δεδομένων ή της ιεράρχησής τους. Ως εκ τούτου χρειάστηκε να ακολουθήσει της εισαγωγής, μια εκτενής προσθαφαίρεση κλάσεων και σχέσεων, ώστε το αποτέλεσμα να μπορεί να δώσει την σημασιολογική διάσταση για την οποία δημιουργήθηκε. Το 1-1 (ένα προς ένα) mapping της βάσης σε οντολογία δεν περιλαμβάνεται σε κάποιο καθολικό πρότυπο και έτσι υπόκειται στον

εκάστοτε διαχειριστή της οντολογίας και της βάσης και εξαρτάται από τους τρόπους χρήσης.

Μετά την δημιουργία της οντολογίας που πληροί τις αρχικές απαιτήσεις σκοπός ήταν να δύναται ο εκάστοτε χρήστης της βάσης δεδομένων να πραγματοποιεί σημασιολογικές επερωτήσεις πάνω στη συγκεκριμένη βάση από οποιοδήποτε μέρος του πλανήτη παρέχεται πρόσβαση στο Διαδίκτυο. Έτσι ήταν απαραίτητο να χρησιμοποιηθεί ένας εξυπηρετητής δικτύου (web server). Για την συγκεκριμένη εφαρμογή επελέγη ο Apache Tomcat που παρέχει υποστήριξη Java. Ταυτόχρονα προκειμένου η δημιουργηθείσα οντολογία να μπορεί να δέχεται σημασιολογικές επερωτήσεις από κάποια λογισμική διαπροσωπεία, πρέπει να χρησιμοποιηθεί λογισμικό εξαγωγής συμπεράσματος (reasoner) στο οποίο θα τίθενται οι ερωτήσεις και αυτό θα τις διαχειρίζεται απαντώντας ανάλογα σε κάθε περίπτωση. Ο reasoner που χρησιμοποιήθηκε για την εφαρμογή αυτή είναι ο Racer. Τέλος το συνολικό σύστημα τέθηκε σε ισχύ και οι σημασιολογικές επερωτήσεις απαντήθηκαν επιτυχώς.

Στο σημείο αυτό θα ήθελα να ευχαριστήσω θερμά τον ερευνητή Γιώργο Στάμου για την πολύτιμη αρωγή και καθοδήγησή του, καθώς και τον επιβλέποντα καθηγητή μου κ.Στέφανο Κόλλια.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ : ΟΝΤΟΛΟΓΙΑ, ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ, ORACLE, ΠΕΡΙΓΡΑΦΙΚΗ ΛΟΓΙΚΗ, SEMANTIC WEB, RDF ,OWL ,PROTÉGÉ, REASONER, RACER

ABSTRACT

The focal point of this dissertation is the attempt to enrich the classic entity-relational databases, with semantic content that does not already reside in them. A modern Database Management System (DBMS) offers a wide variety of procedures in order to maintain data integrity and efficient management. However, it does not provide any semantic characterization to data themselves. The transition, from a clearly human-oriented to a both machine-interpretable and human-understandable way of representing data can be accomplished by the use of Ontologies.

In the attempt to adjoin semantic meaning to plain data in a database or elsewhere, one needs to add or annotate data with a well-defined piece of information. This can be achieved for web sites and portals with the use of RDF (Resource Description Framework) technologies and for network or local databases with the W3C (World Wide Web Consortium) approved OWL (Ontology Web Language) implementation. The way and the standards that need to be followed for this additamend are the major issues of this dissertation.

For the current project a test database was used, in the way that was described by its Oracle Schema for a certain field of interest. Great attention was given to the fact that the test database included all the integrity and data constraints - in their vast majority - that can be found in a database schema. This procedure included a migration of data from the database to an ontology editing and development tool used worldwide, named Protégé 3.0. For the migration, a certain plug-in was used, that allows the transition of data from Oracle to Protégé without any loss of information.

The initially imported structure from the database schema could by no means meet the needs of a semantically useful ontology, since the database itself does not contain any formal description of the ways that data will be used or classified. Thus, after the import of the information in a raw form, a thin grain analysis was adopted via constant addition and subtraction of information, classes and relations. The result needed to fulfil the semantic approach for which it was initially sought. The 1-1 mapping of a database to an ontology is nowhere officially described or followed by a standard procedure.

After the creation of the ontology, that met the requirements originally stated, the next goal of the project was to provide the ability to any user to make semantic queries to

the specific ontology from any place in the world where Internet access is provided. Thus, the use of a web server was vital. For this project Apache Tomcat was chosen, since it provides extended support of Java web applications. In the meantime, the need for a reasoner was apparent, since the intention was to create a software interface that would deal with semantic queries placed by users. The reasoner that was used was Racer (in its educational version) and the query language was RQL (Racer Query Language). A software interface was also needed to help divert the information from a web input tab to a Racer compatible mode. Finally the whole system was tested and the semantic queries where successfully answered.

KEY WORDS: Ontology, Database, Oracle, Description Logics, Semantic Web, RDF, OWL, Protégé, Reasoner, Racer.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΕΧΟΜΕΝΑ....9
ΠΡΟΛΟΓΟΣ....10
ΚΕΦΑΛΑΙΟ 1ο : ΠΕΡΙΓΡΑΦΙΚΗ ΛΟΓΙΚΗ....12
1.1 Εισαγωγή....13
1.2 Από τα Δίκτυα στην Περιγραφική Λογική....16
1.2.1 Ένας λογικός απολογισμός των δικτυακών δομών....18
1.3 Reasoning (Συμπερασματολογία)....21
1.4 Αναπαράσταση Γνώσης στην Περιγραφική Λογική....25
1.4.1 Το TBox (Σώμα Όρων)....27
1.4.2 Το ABox (Σώμα Υποθέσεων ή Σώμα Ισχυρισμών)....29
1.5 Γλώσσες Περιγραφής....31
1.5.1 Η οικογένεια των AL – γλωσσών....33
1.5.2 Οι γλώσσες περιγραφής ως τμήματα της κατηγορηματικής λογικής....35
1.6 Ορολογίες (Terminologies)....36
1.6.1 Τα αξιώματα ορολογιών....36
1.6.2 Ορισμοί....37
ΚΕΦΑΛΑΙΟ 2ο : SEMANTIC WEB....40
2.1 Τι είναι το Semantic Web;....42
2.2 Γιατί χρειαζόμαστε το Semantic Web;....46
2.2.1 Υπερφόρτωση Πληροφοριών (Information Overloading)....46
2.2.2 Stovepipe Συστήματα....47
2.2.3 Φτωχή συνάθροιση περιεχομένων (Poor Content Aggregation)....48
2.3 Οντολογίες....49
2.3.1 Οι ορισμοί οντολογιών....53
2.3.2 Η σύνταξη, η δομή και η σημασιολογία....59
2.3.3 Επίπεδα Αναπαράστασης....65
2.3.4 Η οντολογία και το σημασιολογικό πρόβλημα χαρτογράφησης (mapping)....66
ΚΕΦΑΛΑΙΟ 3ο : ΕΦΑΡΜΟΓΗ....70
3.1 Από το Oracle schema στον σημασιολογικό κόσμο....71
3.2 Protégé....92
3.3 Tomcat Web Server....118
3.4 Racer, the Reasoner....120
Βιβλιογραφία....123

ΠΡΟΛΟΓΟΣ

Μια επανάσταση ξεκινάει... Μια επανάσταση απέναντι σε εκείνους που πιστεύουν ότι το Διαδίκτυο δεν είναι τίποτε παραπάνω από εκατομμύρια χιλιόμετρα χαλκού και συνδέσεις ολοκληρωμένων κυκλωμάτων.Κι όμως...Η πνευματική περιουσία της ανθρωπότητας συγκεντρώνεται σε ακολουθίες δυαδικών ψηφίων, στα δεδομένα, που είναι πια σε θέση να λάβουν μέρος στην κοσμογονία που επιτελείται γύρω τους. Οι αρχιτέκτονες της σύγχρονης λεωφόρου για την παγκόσμια διακίνηση της γνώσης, παύουν να τα μετέρχονται ως άνοα οικοδομικά υλικά. Τα δεδομένα μετατρέπονται από εύθρυπτοι πλίνθοι σε φέροντα οργανισμό, ικανά να παρέχουν ουσιαστική σημασιολογική στήριξη στον δυσθεώρητο πύργο της αναπαράστασης γνώσης. Ο χρόνος μόνο θα δείξει πόσο και αν αυτή η αλλαγή θα επιτρέψει στην ανθρωπότητα να αγγίξει με τα επιτεύγματά της τα επουράνια περνώντας ουσιαστικά σε μια εποχή νοημόνων μηχανών ή θα καταποντιστεί ως άλλος πύργος της Βαβέλ...

Τα μεγαλεία να φοβάσαι ώ ψυχή.
Και τες φιλοδοξίες σου να υπερνικήσεις
αν δεν μπορείς, με δισταγμό και προφυλάξεις
να τες ακολουθείς. Κι όσο εμπροστά προβαίνεις,
τόσο εξεταστική, προσεκτική να είσαι.
.....
(«Μάρτιαι Ειδοί», Κ. Π. Καβάφης, 1911)

Η εργασία αυτή αφιερώνεται εξαιρετικά στους γονείς μου, Σταύρο και Αλίκη για την αμέριστη στήριξη και συμπαράσταση σε όλες τις ως τώρα προσπάθειές μου. Στην γιαγιά-Τζούλια, για την επιμονή και την υπομονή που με δίδαξε να αντιμετωπίζω τις δυσκολίες που προκύπτουν. Στο Τζουλάκι, για την χαρά και την ζωντάνια που αντανακλά και προσφέρει απλόχερα σε κάθε στιγμή της ζωής μου. Στο αγαπημένο Άννα-Μαράκι για το κέφι και την αισιοδοξία που μου εμπνέει άλλα και για την πίστη που μου δείχνει κινητοποιώντας με να συνεχίζω τον αγώνα μέχρι να πετύχω...

ΚΕΦΑΛΑΙΟ 1^{ο} : ΠΕΡΙΓΡΑΦΙΚΗ ΛΟΓΙΚΗ

1.1 Εισαγωγή

Οι στόχοι της έρευνας στο πεδίο της αναπαράστασης γνώσης και στην συμπερασματολογία της ανάκτησής της (reasoning), συνήθως επικεντρώνονται στις μεθόδους δημιουργίας περιγραφών του «κόσμου» με υψηλή ακρίβεια, ώστε να επιτυγχάνεται η ανάπτυξη νοημόνων εφαρμογών που να παρέχουν χρήσιμα αποτελέσματα. Με αυτή την προοπτική, η νοημοσύνη έγκειται στη δυνατότητα του συστήματος να μπορεί να εντοπίζει και να εξάγει υπονοούμενα συμπεράσματα (implicit consequences) από την ρητά αναπαριστάμενη γνώση που του δίδεται. Τέτοια συστήματα χαρακτηρίζονται ως συστήματα βάσης γνώσης (knowledge – based systems).

Οι πρώτες προσεγγίσεις σε θέματα συστημάτων αναπαράστασης γνώσης αναπτύχθηκαν την δεκαετία του 1970 και διακρίνονται σε δύο υποκατηγορίες: στα συστήματα που βασίζονται στους λογικούς φορμαλισμούς, που προήλθαν από την διαίσθηση ότι ο κατηγορηματικός λογισμός μπορεί να χρησιμοποιηθεί χωρίς αμφισημίες για να κατανοεί δεδομένα του «κόσμου» και στα συστήματα που είναι βασισμένα σε μη λογικές αναπαραστάσεις γνώσης. Τα τελευταία συχνά αναπτύσσονταν βασιζόμενα περισσότερο σε τρόπους και ιδέες γνώσης καθαρά ανθρωποκεντρικές – για παράδειγμα οι δικτυακές δομές και οι βασισμένες σε κανόνες αναπαραστάσεις, προήλθαν από πειράματα ανάκτησης της ανθρώπινης μνήμης και μεθόδους επίλυσης προβλημάτων από τον άνθρωπο όπως η επίλυση μαθηματικών προβλημάτων. Παρά το ότι αυτές οι προσεγγίσεις συχνά αναπτύσσονταν για πολύ συγκεκριμένα πεδία αναπαραστάσεων, οι συνεπαγόμενοι φορμαλισμοί όφειλαν να δύνανται να εξυπηρετούν σε κάθε γενική περίπτωση. Με άλλα λόγια, τα μη λογικά συστήματα που προήλθαν από πολύ στενές οδούς σκέψης (όπως για παράδειγμα τα συστήματα παραγωγής) εξελίχθηκαν να αντιμετωπίζονται σαν γενικής χρήσης εργαλεία, που να είναι ικανά να εφαρμόζονται σε διαφορετικά πεδία και σε ποικιλία προβλημάτων.

Από την άλλη πλευρά, με βάση την λογική πρώτης τάξης που παρέχει πολύ ισχυρή και γενική λειτουργικότητα, οι λογικό–στραφείς προσεγγίσεις αποτελούσαν γενικής χρήσεις εργαλεία εξ'αρχής. Με μια λογικά βασισμένη προσέγγιση, η γλώσσα αναπαράστασης είναι συνήθως ένα είδος πρωτοβάθμιου κατηγορηματικού λογισμού (predicate calculus), και η συμπερασματολογία της γνώσης ισοδυναμεί με ικανοποίηση λογικών ακολουθιών. Στις μη λογικές προσεγγίσεις που συνήθως βασίζονται στη χρήση

γραφικών διαπροσωπειών, η γνώση αναπαρίσταται με μέσα όπως ad hoc δομές δεδομένων και η ανάκτηση επιτυγχάνεται με παρόμοιες ad hoc διαδικασίες που διαχειρίζονται τις δομές αυτές. Ανάμεσα στις εξειδικευμένες αυτές εφαρμογές βρίσκουμε τα σημασιολογικά δίκτυα και τα frames. Τα σημασιολογικά δίκτυα αναπτύχθηκαν μετά από την εργασία του Quillian [1967], με στόχο τον χαρακτηρισμό – με δικτυακά σχηματισμένες γνωσιακές δομές – της γνώσης και της συμπερασματολογίας που εφαρμόζεται σε αυτήν από το σύστημα. Παρόμοιοι ήταν και οι στόχοι μεταγενέστερων frame συστημάτων [Minsky, 1981] που βασίζονταν στην ιδέα του frame σαν πρωτότυπου και στην ικανότητα έκφρασης σχέσεων μεταξύ των frames. Παρά τις σημαντικές διαφορές μεταξύ των σημασιολογικών δικτύων και των frames, και τα δύο στις κινητοποιούς γνωσιακές του ιδέες και στα χαρακτηριστικά τους, έχουν μια κοινή βάση. Ουσιαστικά μπορούν και τα δυο να χαρακτηριστούν σαν δικτυακές δομές, όπου η δομή του δικτύου στοχεύει στην αναπαράσταση συνόλων από άτομα και των μεταξύ τους σχέσεων. Συμπερασματικά, μπορούμε να χρησιμοποιήσουμε τον όρο «δικτυακές δομές » (network – based structures), για να αναφερθούμε στα παραστατικά δίκτυα από τα οποία προκύπτουν τα σημασιολογικά δίκτυα και τα frames.

Εξαιτίας της ανθρωποκεντρικής τους προσέγγισης, τα δικτυακά συστήματα θεωρούνταν γενικά πιο κατάλληλα στην εφαρμογή απ’ ότι τα συστήματα που βασίζονταν στις αρχές της λογικής. Δυστυχώς δεν ήταν απόλυτα ικανοποιητικά καθώς συνήθως στερούνταν ενός ακριβούς σημασιολογικού χαρακτηρισμού. Το αποτέλεσμα ήταν να προκύπτουν συστήματα που το καθένα συμπεριφέρονταν διαφορετικά από το άλλο, παρά το ότι σε πολλές περιπτώσεις χρησιμοποιούσαν πανομοιότυπες δομές και συστατικά ακόμα και ίδια ονόματα σχέσεων. Το ερώτημα που ενέσκηψε ήταν προφανές. Ποιος θα ήταν ο τρόπος να ενσωματωθεί σημασιολογία σε παραστατικές δομές – συγκεκριμένα σε σημασιολογικά δίκτυα και σε frames – ώστε εκμεταλλευόμενες την ιεραρχική τους δομή, να επιτρέπουν ταυτόχρονα ευκολία αναπαράστασης και αποδοτικότητα στην ανάκτηση της γνώσης;

Το πρώτο βήμα στην κατεύθυνση αυτή, έγινε με την αναγνώριση ότι στα frames μπορούσε να αποδοθεί σημασιολογία με χρήση της λογικής πρώτης τάξης [Hayes 1979]. Τα βασικά στοιχεία της αναπαράστασης ήταν τα μοναδικά κατηγορήματα (unary predicates), που αντιπροσώπευαν σύνολα ατόμων και τα δυαδικά κατηγορήματα (binary predicates), που αντιπροσώπευαν σχέσεις μεταξύ αυτών των ατόμων. Ωστόσο μια τέτοια προσέγγιση, δεν ήταν δυνατό να συνάδει με τους περιορισμούς που διέπουν τα σημασιολογικά δίκτυα σε σχέση με την λογική. Στην ουσία, παρ’ όλο που η λογική είναι

η φυσική βάση για τον καθορισμό μιας έννοιας σε αυτές τις δομές, διαφαινόταν ότι τα frames και τα σημασιολογικά δίκτυα δεν χρειάζονταν όλους τους μηχανισμούς της λογικής πρώτης τάξης, παρά μόνο τμήμα αυτής [Brachman and Levesque 1985]. Επιπρόσθετα, διαφορετικά χαρακτηριστικά μιας γλώσσας αναπαράστασης, οδηγούν σε διαφορετικά τμήματα της λογικής πρώτης τάξης. Το γεγονός αυτό είχε μια – πλην άλλων – σημαντική συνέπεια. Οδήγησε στην συνειδητοποίηση ότι οι κλασικοί τρόποι ανάκτησης γνώσης που χρησιμοποιήθηκαν σε δόμο – κεντρικές αναπαραστάσεις μπορούσαν να επιτευχθούν με εξειδίκευση στις τεχνικές ανάκτησης, χωρίς απαραίτητα να απαιτούνται αποδείξεις θεωρημάτων της λογικής. Συνεπώς η συμπερασματολογία σε διαφορετικά τμήματα λογικής πρώτης τάξης οδηγεί σε υπολογιστικά προβλήματα διαφορετικής πολυπλοκότητας.

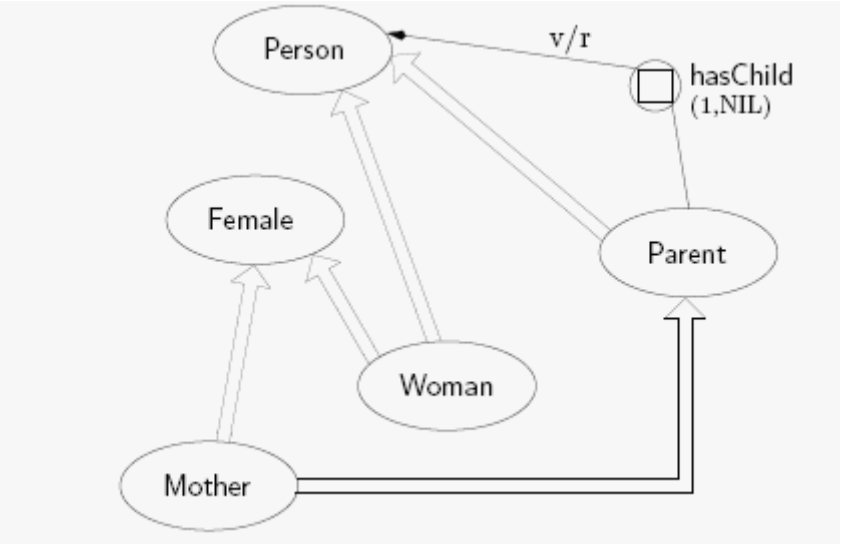
Επακόλουθο αυτής της συνειδητοποίησης ήταν η ανάπτυξη ενός νέου πεδίου έρευνας της Περιγραφικής Λογικής υπό το όνομα «Συστήματα Ορολογιών» (Terminological Systems). Ο λόγος που επικράτησε αυτή η ονομασία ήταν περισσότερο για να επιστήσει το ενδιαφέρον στο γεγονός ότι η γλώσσα αναπαράστασης ήταν εκείνη που καθόριζε την βασική ονοματολογία που υιοθετούνταν για το εκάστοτε πεδίο αναπαράστασης. Αργότερα η έμφαση δόθηκε περισσότερο στους κατασκευαστές (constructors) που διαμόρφωναν τα concepts που αποδεχόταν η γλώσσα. Πιο πρόσφατα και αφότου η προσοχή εστιάστηκε στις ιδιότητες των υποκείμενων λογικών συστημάτων των εφαρμογών επικράτησε ο όρος Περιγραφική Λογική.

Κλείνοντας την σύντομη αυτή εισαγωγή, αξίζει να αναφερθεί ότι η Περιγραφική Λογική έχει χρησιμοποιηθεί για να καλύψει τόσο το θεωρητικό υπόβαθρο των συστημάτων αναπαράστασης γνώσης όσο και το κομμάτι της εφαρμογής αυτών και της ανάπτυξης εφαρμογών σε πληθώρα πεδίων. Η όλη προσπάθεια χαρακτηρίζεται αναμφίβολα επιτυχής. Το κλειδί για την επιτυχία αποτέλεσε η μεθοδολογία της στενής συνεργασίας μεταξύ θεωρίας και εφαρμογής. Ταυτόχρονα έχουν αναπτυχθεί αρκετά συστήματα βασισμένα στην Περιγραφική Λογική που παρέχουν μια ποικιλία σημασιολογικών φορμαλισμών με διαφορετική εκφραστική ικανότητα για διάφορους τομείς έρευνας (όπως η επεξεργασία φυσικής γλώσσας, η διαμόρφωση τεχνικών προϊόντων και οι βάσεις δεδομένων). Επιπρόσθετα οι μαθηματικά αποδεδειγμένες και οι υπολογιστικά ανεπτυγμένες μέθοδοι ανάκτησης γνώσης (σε επίπεδο απόφασης και πολυπλοκότητας) διαφόρων σημασιολογικών φορμαλισμών εξερευνήθηκαν σε βάθος. Έτσι τα αποτελέσματα αυτών των ερευνών οδήγησαν στο σχεδιασμό νέων συστημάτων και στην περαιτέρω εξέλιξη του τομέα.

1.2 Από τα Δίκτυα στην Περιγραφική Λογική

Προκειμένου να παρασχεθεί κάποια διαίσθηση για τις ιδέες πίσω από τις αναπαραστάσεις της γνώσης με δίκτυα, από την άποψη ενός γενικού δικτύου, αποφεύγουμε τις αναφορές σε οποιοδήποτε ιδιαίτερο σύστημα. Τα στοιχεία ενός δικτύου είναι οι κόμβοι και οι συνδέσεις. Κυριολεκτικά, οι κόμβοι χρησιμοποιούνται για να χαρακτηρίσουν τις έννοιες, δηλαδή, τα σύνολα ή τις κατηγορίες μεμονωμένων αντικειμένων, και οι συνδέσεις χρησιμοποιούνται για να χαρακτηρίσουν τις σχέσεις μεταξύ τους. Σε μερικές περιπτώσεις, οι πιό σύνθετες σχέσεις αντιπροσωπεύονται αυτούσιες ως κόμβοι. Αυτοί διακρίνονται ωστόσο από τους κόμβους που αντιπροσωπεύουν τις έννοιες. Επιπλέον, οι έννοιες μπορούν να έχουν απλές ιδιότητες οι οποίες είναι συνδεμένες με τους αντίστοιχους κόμβους. Αξίζει μνείας το γεγονός ότι σε αρκετά από τα πρώτα δίκτυα, τόσο τα μεμονωμένα αντικείμενα όσο και οι έννοιες αντιπροσωπεύονταν από κόμβους. Εδώ, εντούτοις, περιορίζουμε την προσοχή μας στη γνώση για τις έννοιες και τις σχέσεις τους, αναβάλλοντας για τώρα την επεξεργασία της γνώσης για τα συγκεκριμένα άτομα.

Εξετάστε ένα απλό παράδειγμα, του οποίου εικονογραφική αντιπροσώπευση δίνεται στο σχήμα 1. 1, το οποίο αντιπροσωπεύει τη γνώση σχετικά με τα πρόσωπα, τους γονείς, τα παιδιά, κ. λπ. . . . Η δομή στην εικόνα αναφέρεται επίσης ως ορολογία (terminology), και προορίζεται να αντιπροσωπεύσει τη γενικότητα/την ιδιομορφία των σχετικών εννοιών. Παραδείγματος χάριν η σύνδεση μεταξύ της *μητέρας* και του *γονέα* λέει ότι «οι μητέρες είναι γονείς». Αυτό καλείται ως «is-A» σχέση.



Εικόνα 1.1

Η is-A σχέση καθορίζει μια ιεραρχία πέρα από τις έννοιες και παρέχει τη βάση για τη «κληρονομιά των ιδιοτήτων»: όταν μια έννοια είναι πιό συγκεκριμένη από κάποια άλλη έννοια, κληρονομεί τις ιδιότητες της γενικότερης. Παραδείγματος χάριν, εάν ένα πρόσωπο έχει μια ηλικία, κατόπιν μια μητέρα έχει μια ηλικία, επίσης. Αυτό είναι ο χαρακτηριστικός καθορισμός των αποκαλούμενων (μονοτονικών) δικτύων κληρονομιάς [Brachman, 1979].

Ένα χαρακτηριστικό γνώρισμα της Περιγραφικής Λογικής είναι η δυνατότητά της να αντιπροσωπεύει άλλα είδη σχέσεων που μπορούν να ενυπάρχουν μεταξύ των εννοιών, πέρα από τις is-A σχέσεις. Παραδείγματος χάριν, στο σχήμα 1. 1, που ακολουθεί τη σημείωση [Brachman και Schmolze, 1985], η έννοια του *γονέα* έχει μια ιδιότητα που καλείται «ρόλος». Εκφράζεται δε με μια σύνδεση από την έννοια (γονέας) σε έναν κόμβο και προκύπτει έτσι ο επονομαζόμενος ρόλος *έχειΠαιδί*. Ο ρόλος διαθέτει έναν «περιορισμό αξίας», που δείχνεται με την συντομογραφία v/r (εκ του value restriction) και ο οποίος εκφράζει έναν περιορισμό στο πεδίο των τύπων αντικειμένων που μπορούν να «γεμίσουν» εκείνο τον ρόλο. Επιπλέον, ο κόμβος εκφράζει έναν περιορισμό αριθμού όπως (1, ΜΗΔΕΝ), όπου ο πρώτος αριθμός είναι ένα κάτω φράγμα του αριθμού των παιδιών και το δεύτερο στοιχείο είναι το άνω φράγμα. Το ΜΗΔΕΝ αντιστοιχεί στο άπειρο. Συνολικά, η αναπαράσταση της έννοιας του γονέα μπορεί εδώ να διαβαστεί ως εξής: «ένας *γονέας* είναι ένα πρόσωπο που έχει τουλάχιστον ένα παιδί, και όλα τα παιδιά του/της είναι πρόσωπα».

Οι σχέσεις αυτού του είδους κληρονομούνται από τις έννοιες στις υπό-έννοιές τους. Παραδείγματος χάριν, η έννοια *μητέρα*, δηλ. , μια θηλυκή γονέας, είναι πιό συγκεκριμένος

απόγονος τόσο του *θηλυκού* όσο και της έννοιας *γονέας*, και κατά συνέπεια κληρονομεί από το *γονέα* τη σύνδεση με το *πρόσωπο* μέσω του ρόλου *έχειΠαιδί* με άλλα λόγια, η *μητέρα* κληρονομεί τον περιορισμό στο ρόλο *έχειΠαιδί* της από το *γονέα*.

Παρατηρήστε ότι μπορούν να υπάρξουν υπονοούμενες σχέσεις μεταξύ των εννοιών. Παραδείγματος χάριν, εάν καθορίζουμε τη *γυναίκα* ως έννοια ενός θηλυκού προσώπου, συμβαίνει ότι κάθε *μητέρα* είναι *γυναίκα*. Είναι στόχος του συστήματος αναπαράστασης γνώσης να βρεθούν οι υπονοούμενες σχέσεις όπως αυτή (πολλές είναι πιό σύνθετες από αυτή). Ουσιαστικά, τέτοια συμπεράσματα προκύπτουν μέσα από τις ιδιότητες του δικτύου. Σε αυτήν την περίπτωση είναι εμφανές ότι και η *μητέρα* και η *γυναίκα* συνδέονται και με το *θηλυκό* και με το *πρόσωπο*, αλλά η πορεία από τη *μητέρα* στο *πρόσωπο* περιλαμβάνει τον κόμβο της έννοιας *γονέας*, που είναι πιό συγκεκριμένη από το πρόσωπο, επιτρέποντας κατά συνέπεια σε μας να καταλήξουμε στο συμπέρασμα ότι η *μητέρα* είναι πιό συγκεκριμένη από το *πρόσωπο*.

Εντούτοις, όσο πιό σύνθετες γίνονται οι σχέσεις που υπάρχουν μεταξύ των εννοιών, τόσο δυσκολότερο γίνεται να δοθεί ένας ακριβής χαρακτηρισμός του είδους των σχέσεων που μπορουν να υπολογιστούν. Είναι επίσης σημαντικό να επιτευχθεί το παραπάνω χωρίς αποτυχία στην αναγνώριση μερικών σχέσεων ή χωρίς να δοθούν λανθασμένες απαντήσεις.

1.2.1 Ένας λογικός απολογισμός των δικτυακών δομών

Βασισμένα στις προαναφερθείσες ιδέες, διάφορα συστήματα εφαρμόστηκαν και χρησιμοποιήθηκαν σε πολλά είδη εφαρμογών. Κατά συνέπεια, προέκυψε η ανάγκη για έναν ακριβή χαρακτηρισμό των δομών που χρησιμοποιήθηκαν στις αναπαραστάσεις αλλά και του συνόλου συμπερασμάτων που θα μπορούσαν να προέλθουν από εκείνες τις δομές.

Ένας ακριβής χαρακτηρισμός ενός δικτύου μπορεί να δοθεί με τον καθορισμό μιας γλώσσας για τα στοιχεία της δομής και με την παροχή μιας ερμηνείας για τις συμβολοακολουθίες (strings) της γλώσσας. Έτσι, ενώ η σύνταξη μπορεί να έχει τις διαφορετικές αποχρώσεις, η σημασιολογία δίνεται χαρακτηριστικά ως σημασιολογία Tarsky ύφους.

Για τη σύνταξη εισάγουμε ένα είδος αφηρημένης γλώσσας, η οποία μοιάζει με άλλους λογικούς φορμαλισμούς. Το βασικό βήμα της κατασκευής παρέχεται από δύο διαχωρισμένα αλφάβητα συμβόλων που χρησιμοποιούνται για να δείξουν :

1. τις ατομικές έννοιες, που υποδεικνύονται από μοναδιαίο σύμβολα κατηγορήματος,

2. και τους ατομικούς ρόλους, που υποδεικνύονται από τα δυαδικά σύμβολα κατηγορήματος.

Τα τελευταία χρησιμοποιούνται για να εκφράσουν τις σχέσεις μεταξύ των εννοιών**.**

Οι όροι χτίζονται από τα βασικά σύμβολα, χρησιμοποιώντας διάφορα είδη «κατασκευαστών». Παραδείγματος χάριν, η τομή των εννοιών, που δείχνεται ως $C \sqcap D$, χρησιμοποιείται για να περιορίσει το σύνολο των υπό εξέταση ατόμων σε εκείνα που ανήκουν και στο C και στο D. Είναι εμφανές ότι, στη σύνταξη της Περιγραφικής Λογικής, οι εκφράσεις των εννοιών δεν συνοδεύονται από μεταβλητές. Στην πραγματικότητα, μια έκφραση έννοιας δείχνει το σύνολο όλων των ατόμων που ικανοποιούν τις ιδιότητες που περιγράφονται με την έκφραση. Επομένως, το $C \sqcap D$ μπορεί να θεωρηθεί ως η πρόταση πρωτοβάθμιας λογικής $C(x) \wedge D(x)$, όπου η μεταβλητή αντιπροσωπεύει όλα τα άτομα στο πεδίο ορισμού και το $C(x)$ ισχύει για εκείνα τα άτομα που ανήκουν στην έννοια *C*.

Τα βασικά χαρακτηριστικά γνωρίσματα της Περιγραφικής Λογικής αποτελούν τα κατασκευάσματα για την καθιέρωση των σχέσεων μεταξύ των εννοιών. Οι βασικοί είναι περιορισμοί αξίας. Παραδείγματος χάριν, ένας περιορισμός αξίας, που γράφεται ως $\forall R.C$, απαιτεί ότι όλα τα άτομα που είναι στη σχέση *R* με την έννοια που περιγράφεται να ανήκουν και στην έννοια *C* (συγκεκριμένα, είναι όλα τα άτομα που είναι στη σχέση R με ένα άτομο που περιγράφεται από την υπό ερώτηση έννοια τα οποία - τα ίδια - περιγράφονται από την C).

Όσον αφορά στη σημασιολογία, στις έννοιες δίνεται μια σύνολο-θεωρητική ερμηνεία: μια έννοια ερμηνεύεται ως σύνολο ατόμων και οι ρόλοι ερμηνεύονται ως σύνολα ζευγαριών των ατόμων. Η περιοχή της ερμηνείας μπορεί να επιλεχτεί αυθαίρετα, και μπορεί να είναι άπειρη. Το μη-πεπερασμένο της περιοχής και η υπόθεση ανοικτού κόσμου (open – world assumption) διακρίνουν τα χαρακτηριστικά γνωρίσματα της Περιγραφικής Λογικής από γλώσσες μοντελοποίησης που αναπτύσσονται στη μελέτη των βάσεων δεδομένων.

Οι ατομικές έννοιες ερμηνεύονται έτσι ως υποσύνολα της περιοχής ερμηνείας (interpretation domain), ενώ η σημασιολογία των άλλων κατασκευασμάτων διευκρινίζεται με τον καθορισμό του συνόλου ατόμων που δημιουργούνται από κάθε κατασκεύασμα. Παραδείγματος χάριν, η έννοια $C \sqcap D$ είναι το σύνολο ατόμων που λαμβάνεται με την τομή των ατόμων που ανήκουν στο *C* και στο *D*, αντίστοιχα. Ομοίως, η ερμηνεία $\forall R.C$ είναι το σύνολο ατόμων που είναι στη σχέση *R* με τα άτομα που ανήκουν στο σύνολο που δείχνεται από την έννοια *C*.

Για παράδειγμα, υποθέστε ότι το *θηλυκό*, το *πρόσωπο*, και η *γυναίκα* είναι ατομικές έννοιες και ότι τα *έχειΠαιδί* και *έχειΘηλυκόΣυγγενή* είναι ατομικοί ρόλοι. Χρησιμοποιώντας τους τελεστές τομή, ένωση και συμπλήρωμα για τις έννοιες, που ερμηνεύονται ως τελεστές συνόλων, μπορούμε να περιγράψουμε την έννοια «των προσώπων που δεν είναι θηλυκά» και την έννοια «άτομα που είναι θηλυκά ή αρσενικά» με τις εκφράσεις :

$$Ατομο \sqcap \neg Θηλυκο \quad \text{και} \quad Αρσενικο \sqcup Θηλυκο$$

Εκτός από τα παραπάνω, σημαντικό είναι να αναφερθούμε και στους ποσοτικοποιημένους περιορισμούς ρόλου και στη συνέχεια, σε αυτό που καλούμε «περιορισμούς αριθμού». Οι περισσότερες γλώσσες παρέχουν (καθολική) υπαρξιακή ποσοτικοποίηση (full existential quantification) και αξιακούς περιορισμούς (value restrictions) που επιτρέπουν την περιγραφή, παραδείγματος χάριν, της έννοιας «των ατόμων που έχουν ένα κορίτσι» ως $\exists \text{Ε}χει\Pi αιδι.\text{Α}τομο$, και της έννοιας «όλων των ατόμων των οποίων τα παιδιά είναι θηλυκά» από την έκφραση $\forall εχει\Pi αιδι.\Theta ηλυκο$. Προκειμένου να διακριθεί η λειτουργία κάθε έννοιας στη σχέση, το μεμονωμένο αντικείμενο που αντιστοιχεί στο δεύτερο όρισμα του ρόλου - που αντιμετωπίζεται ως δυαδικό κατηγόρημα - καλείται υλικό πληρώσεως ρόλου. Στις ανωτέρω εκφράσεις, που περιγράφουν τις ιδιότητες των γονέων που έχουν τα κορίτσια, τα μεμονωμένα άτομα που ανήκουν στην έννοια θηλυκό είναι τα υλικά πληρώσεως του ρόλου *εχειΠαιδί*.

Οι υπαρξιακές ποσοτικοποιήσεις και οι περιορισμοί αξίας προορίζονται έτσι να χαρακτηρίσουν τις σχέσεις μεταξύ των εννοιών. Στην πραγματικότητα, η σύνδεση ρόλου μεταξύ του γονέα και του προσώπου στην εικόνα 1. 1 μπορεί να εκφραστεί από το :

$$\exists ΕχειΠαιδι.Ατομο \sqcup \forall εχειΠαιδι.Ατομο$$

Μια τέτοια έκφραση επομένως χαρακτηρίζει την έννοια του γονέα ως σύνολο ατόμων που έχουν τουλάχιστον ένα υλικό πληρώσεως του ρόλου *εχειΠαιδι* που ανήκει στην έννοια πρόσωπο. Επιπλέον, κάθε υλικό πληρώσεως του ρόλου *εχειΠαιδί* πρέπει να είναι πρόσωπο.

Τέλος, παρατηρήστε ότι στους ποσοτικοποιημένους περιορισμούς ρόλου η μεταβλητή που ποσολογείται δεν αναφέρεται ρητά. Η αντίστοιχη πρόταση στην πρώτης τάξης λογικής είναι $\forall y.R(x,y) \supset C(y)$, όπου το χ είναι πάλι μια ελεύθερη μεταβλητή μέσα στο πεδίο ερμηνείας.

Ένα άλλο σημαντικό είδος περιορισμού ρόλου δίνεται από τους περιορισμούς αριθμού, οι οποίοι περιορίζουν τον αριθμό στοιχείων του συνόλου υλικών πληρώσεως των ρόλων. Παραδείγματος χάριν, η έννοια:

$$(\geq 3 \mathit{εχειΠαιδι}) \sqcap (\leq \mathit{εχειΘηλυκουςΣυγγενεις})$$

αντιπροσωπεύει την έννοια «των ατόμων που έχουν τουλάχιστον τρία παιδιά και το πολύ δύο θηλυκούς συγγενείς». Οι περιορισμοί αριθμού αντιμετωπίζονται μερικές φορές ως ιδιαίτερο χαρακτηριστικό γνώρισμα της Περιγραφικής Λογικής, αν και μπορεί να βρεθούν μερικά παρόμοια κατασκευάσματα σε γλώσσες μοντελοποίησης βάσεων δεδομένων (ειδικότερα στα πρότυπα οντοτήτων-συσχετίσεων).

Πέρα από τα «κατασκευάσματα» για την διαμόρφωση εκφράσεων εννοιών, η Περιγραφική Λογική παρέχει «κατασκευαστές» και για τους ρόλους, οι οποίοι μπορούν, παραδείγματος χάριν, να καθιερώσουν τις ιεραρχίες ρόλου. Εντούτοις, η χρήση των εκφράσεων ρόλου περιορίζεται γενικά στην έκφραση των σχέσεων μεταξύ των εννοιών. Η τομή των ρόλων είναι ένα παράδειγμα ενός κατασκευαστή ρόλου. Διαισθητικά, το $\mathit{εχειΠαιδι} \sqcap \mathit{εχειΘηλυκουςΣυγγενεις}$ παράγει το ρόλο *εχειΚόρη*, έτσι ώστε το :

$$\mathit{Γυναικα} \sqcap 2 \leq (\mathit{εχειΠαιδι} \sqcap \mathit{εχειΘηλυκουςΣυγγενεις})$$

δείχνει την έννοια «μιας γυναίκας που έχει το πολύ 2 κόρες».

1.3 Reasoning (Συμπερασματολογία)

Το βασικό ζητούμενο στην έκφραση εννοιών στην Περιγραφική Λογική είναι η υπαγωγή (subsumption), που συμβολίζεται ως $C \sqsubseteq D$. Η εύρεση της σχέσης υπαγωγής είναι το πρόβλημα προς έλεγχο, δηλαδή εάν η έννοια που δείχνεται από το D (ο υπάγων - subsumer) θεωρείται γενικότερη από αυτό που δείχνεται από το C (ο υπαγόμενος - subsumee). Με άλλα λόγια, η υπαγωγή ελέγχει εάν η πρώτη έννοια δείχνει πάντα ένα υποσύνολο του συνόλου που δείχνεται από την δεύτερη.

Έστω ότι κάποιος ενδιαφέρεται για το εάν $\Gamma\mathit{υναικα} \sqsubseteq \mathrm{M}\mathit{ητερα}$. Προκειμένου να ελεγχθεί αυτό το είδος σχέσης, πρέπει να ληφθούν υπόψη οι σχέσεις που καθορίζονται στην ορολογία. Υπό τους κατάλληλους περιορισμούς, μπορεί να ενσωματωθεί τέτοια γνώση άμεσα στις εκφράσεις εννοιών, κάνοντας κατά συνέπεια την υπαγωγή - subsumption των εκφράσεων έννοιας το βασικό στόχο του συλλογισμού (reasoning).

Ένα άλλο ενδιαφέρον γεγονός στις εκφράσεις έννοιας είναι η ικανοποιησιμότητα (satisfiability) της έννοιας. Τούτο σημαίνει ότι μια έννοια είναι ικανοποιήσιμη όταν δεν μπορεί να υπαχθεί στην κενή έννοια.

Αν και η σημασία των εννοιών ήταν διευκρινισμένη ήδη με την λογική σημασιολογία, ο σχεδιασμός των διαδικασιών συμπεράσματος στην Περιγραφική Λογική επηρεάστηκε για αρκετό διάστημα από την παράδοση των σημασιολογικών δικτύων, όπου οι έννοιες αντιμετωπίσθηκαν ως κόμβοι και οι συνδέσεις ως ρόλοι σε ένα δίκτυο. Η υπαγωγή μεταξύ των εννοιών αναγνωρίστηκε ως η βασική επιδίωξη και η πρωταρχική ιδέα των πιό πρόωρων αλγορίθμων υπαγωγής ήταν να μετασχηματιστούν δύο επιλεγμένες έννοιες στις γραφικές τους αναπαραστάσεις προκειμένου να εξετασθεί εάν κάποια θα μπορούσε να ενσωματωθεί στην άλλη. Η ενσωματώσα γραφική παράσταση θα αντιστοιχούσε στη γενικότερη έννοια (το subsumer) [Lipkis, 1982]. Αυτή η μέθοδος καλείται δομική σύγκριση (structural comparison), και η σχέση μεταξύ των εννοιών που υπολογίζονται καλείται δομική υπαγωγή (structural subsumption). Εντούτοις, μια προσεκτική ανάλυση των αλγορίθμων για την δομική υπαγωγή δείχνει ότι είναι ικανοί (sound), αλλά όχι πάντα πλήρεις (complete) από την άποψη της λογικής σημασιολογίας: όποτε επιστρέφουν «ναι» η απάντηση είναι σωστή, αλλά όταν αυτοί απαντούν «όχι» η απάντηση μπορεί να είναι ανακριβής. Με άλλα λόγια, η δομική υπαγωγή είναι γενικά πιό αδύνατη από τη λογική υπαγωγή. Η ανάγκη για τους πλήρεις αλγορίθμους υπαγωγής παρακινείται από το γεγονός ότι στη χρήση των συστημάτων αναπαράστασης γνώσης είναι συχνά απαραίτητο να υπάρξει μια εγγύηση ότι το σύστημα δεν έχει αποτύχει στην επαλήθευση της υπαγωγής. Συνεπώς, οι νέοι αλγόριθμοι για την υπαγωγή που έχουν επινοηθεί δεν είναι πλέον βασισμένοι σε αναπαράσταση δικτύων, και αποδεικνύεται ότι αυτοί είναι πλήρεις. Τέτοιοι αλγόριθμοι έχουν αναπτυχθεί με την επεξεργασία των κλασσικών δεδομένων για τον παραγωγικό συλλογισμό στα υποσύνολα DL της λογικής πρώτης τάξης, όπως γίνονται για την ανάλυση μητρών (tableu calculi) από τους Schmidt-Schauss και Smolka [1991], και επίσης με πιό εξειδικευμένες μεθόδους.

Στο δημοσιευμένο έγγραφο “the Tractability of Subsumption in Frame-Based Description Languages”, των Brachman και Levesque [1984], υποστηρίχθηκε ότι υπάρχει ένα ισοζύγιο (tradeoff) μεταξύ της εκφραστικότητας μιας γλώσσας αναπαράστασης και της δυσκολίας συμπερασματολογίας (reasoning) για τις αναπαραστάσεις που χτίζονται από την γλώσσα. Με άλλα λόγια, όσο πιό εκφραστική η γλώσσα, τόσο σκληρότερος ο συλλογισμός. Παρείχαν επίσης ένα πρώτο παράδειγμα αυτού του ισοζυγίου με την ανάλυση της γλώσσας $\mathcal{FL}^-$ (frame language), η οποία περιλάμβανε την τομή, τους

περιορισμούς αξίας και μια απλή μορφή υπαρξιακού προσδιορισμού της ποσότητας. Συμπερασματικά έδειξαν ότι για μια τέτοια γλώσσα, το πρόβλημα subsumption μπορούσε να λυθεί σε *πολυωνυμικό χρόνο*, ενώ προσθέτοντας ένα κατασκεύασμα αποκαλούμενο περιορισμό ρόλου στη γλώσσα καταντά το subsumption να είναι ένα *coNP-hard* πρόβλημα (η εκτεταμένη γλώσσα κλήθηκε $\mathcal{FL}$).

4. Expressiveness and tractability

It appears that we have run into a serious difficulty in trying to develop a KR service that calculates what is implicit in a KB and yet does so in a reasonable amount of time. One option we have not yet considered, however, is to *limit* what can be in the KB so that its implications are more manageable computationally. Indeed, as we will demonstrate in this section, much of the research in KR can be construed as trading expressiveness in a representation language for a more tractable form of inference. Moreover, unlike the restricted dialects of FOL typical of those analyzed in the logic and computer science literatures (e.g., in terms of nestings of quantifiers), the languages considered here have at least proven themselves quite useful in practice, however contrived they may appear on the surface.

Εικόνα 1.2 [Απόσπασμα από το κείμενο των Brachman και Levesque Expressiveness and Tractability in Knowledge Representation Systems]

Το έγγραφο από Brachman και Levesque εισήγαγε τουλάχιστον δύο νέες ιδέες:

1. «την αποδοτικότητα της συλλογιστικής» για τις δομές γνώσης, η οποία μπορεί να μελετηθεί χρησιμοποιώντας τα εργαλεία της υπολογιστικής θεωρίας πολυπλοκότητας και ότι
2. Οι διαφορετικοί συνδυασμοί κατασκευαστών μπορούν να δημιουργήσουν γλώσσες με διαφορετικές υπολογιστικές ιδιότητες.

Μια άμεση συνέπεια των ανωτέρω παρατηρήσεων είναι ότι μπορεί να μελετηθεί τυπικά και μεθοδικά το ισοζύγιο μεταξύ της υπολογιστικής πολυπλοκότητας του συλλογισμού και της εκφραστικότητας της γλώσσας, το οποίο καθορίζεται από την προσθήκη ή μη των κατασκευαστών που προστίθενται στη γλώσσα. Μετά από το αρχικό κείμενο, διάφορα αποτελέσματα επιτεύχθηκαν για το ισοζύγιο των γλωσσών εννοιών και αυτά τα αποτελέσματα μας επιτρέπουν μια αρκετά πλήρη εικόνα της πολυπλοκότητας του συλλογισμού για μια ευρεία κατηγορία γλωσσών. Επιπλέον, το πρόβλημα για την εύρεση

του βέλτιστου ισοζυγίου, δηλαδή για τις πιό εκφραστικές επεκτάσεις της $\mathcal{FL}^-$ αναφορικά με ένα δεδομένο σύνολο κατασκευαστών που να διατηρούν το πολυώνυμικο subsumption, έχει μελετηθεί εκτενώς [Donini et Al, 1991b,1999]. Μια από τις υποθέσεις που κρύβονται κάτω από αυτήν την γραμμή έρευνας είναι να χρησιμοποιηθεί η worst-case πολυπλοκότητα ως μέτρο της αποδοτικότητας του συλλογισμού στην Περιγραφική Λογική (και γενικότερα στους φορμαλισμούς αντιπροσώπευσης γνώσης). Μια τέτοια υπόθεση έχει επικριθεί (παραδείγματος χάριν [Doyle και Patil, 1991]) ως μη επαρκής να χαρακτηρίσει την απόδοση συστημάτων ή να αποτελέσει την κοινώς αποκαλούμενη συμπεριφορά average-case. Ενώ αυτή η παρατήρηση προτείνει ότι η υπολογιστική πολυπλοκότητα μπορεί μόνη της να μην είναι ικανοποιητική για να αντιμετωπίσει τα ζητήματα απόδοσης, η έρευνα για την υπολογιστική πολυπλοκότητα του συλλογισμού στην Περιγραφική Λογική έχει σίγουρα οδηγήσει σε μια πολύ βαθύτερη κατανόηση των προβλημάτων που προκύπτουν στην εφαρμογή των εργαλείων συλλογισμού.

Καταρχήν η μελέτη της υπολογιστικής πολυπλοκότητας του συλλογισμού στην Περιγραφική Λογική έχει οδηγήσει σε μια σαφή κατανόηση των ιδιοτήτων των γλωσσικών κατασκευαστών και της αλληλεπίδρασής τους. Αυτό δεν είναι μόνο πολύτιμο από μια θεωρητική άποψη, αλλά κυρίως επειδή δίνει τη διορατικότητα στο σχεδιαστή των διαδικασιών αφαίρεσης-υπαγωγής, να γνωρίζει με σαφήνεια τις επιπτώσεις της προσθήκης συγκεκριμένων γλωσσικών κατασκευαστών και των συνδυασμών τους (που είναι δύσκολο να εξεταστούν), καθώς επίσης και τις γενικές μεθόδους αντιμετώπισης.

Αφετέρου, τα αποτελέσματα πολυπλοκότητας έχουν επιτευχθεί με την εκμετάλλευση μιας γενικής τεχνικής για έλεγχο ικανοποιησιμότητας στις γλώσσες, η οποία στηρίζεται σε μια μορφή μαθηματικής ανάλυσης [Schmidt - Schauss και Smolka, 1991]. Μια τέτοια τεχνική έχει αποδειχθεί εξαιρετικά χρήσιμη για την μελέτη τόσο της ακρίβειας όσο και της πολυπλοκότητας των αλγορίθμων. Πιό συγκεκριμένα, η τεχνική αυτή παρέχει ένα αλγοριθμικό πλαίσιο που είναι παραμετρικό όσον αφορά τα γλωσσικά κατασκευάσματα. Οι αλγόριθμοι για την ικανοποιησιμότητα και την υπαγωγή έννοιας που λαμβάνονται κατ' αυτό τον τρόπο έχουν επίσης οδηγήσει στις πρακτικές υλοποιήσεις της εφαρμογής των έξυπνων στρατηγικών ελέγχου και των τεχνικών βελτιστοποίησης. Τα πιό πρόσφατα συστήματα αντιπροσώπευσης γνώσης βασισμένα στην Περιγραφική Λογική υιοθετούν την μαθηματική ανάλυση [Horrocks, 1998b].

Τρίτον, η ανάλυση των παθολογικών περιπτώσεων σε αυτό το επίσημο πλαίσιο έχει οδηγήσει στην ανακάλυψη της μη πληρότητας στους αλγορίθμους που αναπτύσσονται για τα εφαρμοσμένα συστήματα. Αυτό επίσης συνεπώς έχει αποδειχθεί χρήσιμο στον

καθορισμό των κατάλληλων συνόλων δοκιμής για την επαλήθευση των εφαρμογών. Παραδείγματος χάριν, η σύγκριση των εφαρμοσμένων συστημάτων ([Baader et Al, 1992b Heinsohn et Al, 1992]) έχει ωφεληθεί πολύ από τα αποτελέσματα της ανάλυσης πολυπλοκότητας.

Αφότου αναλύθηκε λεπτομερώς το ισοζύγιο μεταξύ της εκφραστικότητας και της ευπείθειας του συλλογισμού και κατόπιν εκτενούς πειραματισμού με την δυνατότητα εφαρμογής των αντίστοιχων τεχνικών συμπεράσματος, υπήρξε μια μετατόπιση της εστίασης στη θεωρητική έρευνα για την συμπερασματολογία στην Περιγραφική Λογική. Το ενδιαφέρον αυξήθηκε στο συσχετισμό μεταξύ της Περιγραφικής Λογικής και των γλωσσών μοντελοποίησης που χρησιμοποιήθηκαν στη διαχείριση βάσεων δεδομένων. Επιπλέον, η ανακάλυψη των αυστηρών σχέσεων με εκφραστικές λογικές υποκίνησε τη μελέτη της αποκαλούμενης *πολύ εκφραστικής* Περιγραφικής Λογικής. Αυτές οι γλώσσες, εκτός από την αναγνώριση των πολύ γενικών μηχανισμών για τις έννοιες (παραδείγματος χάριν κυκλικοί ορισμοί), παρέχουν ένα πλουσιότερο σύνολο κατασκευαστών διαμόρφωσης εννοιών και κατασκευαστών διαμόρφωσης σύνθετων εκφράσεων ρόλων. Για αυτές τις γλώσσες, η εκφραστικότητα είναι τόσο μεγάλη ώστε η νέα πρόκληση έγινε να εμπλουτίζεται διαρκώς η γλώσσα διατηρώντας ταυτόχρονα την ικανότητα απόκρισης (decidability) του κάθε συλλογισμού. Αξίζει τέλος να σημειωθεί ότι αυτή η νέα κατεύθυνση της θεωρητικής έρευνας, συνοδεύθηκε από μια αντίστοιχη μετατόπιση στην εφαρμογή των συστημάτων αναπαράστασης γνώσης που βασίζονται στις πολύ εκφραστικές γλώσσες Περιγραφικής Λογικής.

1.4 Αναπαράσταση Γνώσης στην Περιγραφική Λογική

Παραπάνω έγινε μνεία μιας βασκής γλώσσας αναπαράστασης εννοιών Περιγραφικής Λογικής μαζί με κάποιες σημαντικές τεχνικές συλλογισμού. Ο στόχος μας είναι τώρα να επεξηγήσουμε πώς η Περιγραφική Λογική μπορεί να είναι χρήσιμη στο σχεδιασμό των βασισμένων στη γνώση εφαρμογών, δηλαδή, πώς μια γλώσσα DL χρησιμοποιείται σε ένα σύστημα αναπαράστασης γνώσης που παρέχει μια γλώσσα για τον καθορισμό μιας βάσης γνώσεων και τα εργαλεία για να εξάγει τα συμπεράσματα από αυτήν. Η υλοποίηση των συστημάτων γνώσης περιλαμβάνει δύο αρχικές πτυχές. Η πρώτη συνίσταται στην παροχή

ενός ακριβούς χαρακτηρισμού μιας βάσης γνώσεων. Αυτό περιλαμβάνει έναν ακριβή χαρακτηρισμό του είδους της γνώσης που ορίζεται στο σύστημα καθώς επίσης και σαφή καθορισμό των συλλογιστικων υπηρεσιών που το σύστημα χρειάζεται να παρέχει – δηλαδή το είδος των ερωτήσεων που το σύστημα πρέπει να είναι σε θέση να απαντήσει. Η δεύτερη πτυχή συνίσταται στην παροχή ενός πλούσιου περιβάλλοντος ανάπτυξης όπου ο χρήστης μπορεί να ωφεληθεί από τις διαφορετικές υπηρεσίες που μπορούν να καταστήσουν την αλληλεπίδρασή του με το σύστημα αποτελεσματικότερη. Ουσιαστικά πρόκειται για μια συνολική αξιολόγηση της λογικής δομής της βάσης γνώσεων και των υπηρεσιών που δύναται να προσφέρει βάσει του σχεδίου των συστημάτων και των εργαλείων για την ανάπτυξη των εφαρμογών από τα οποία συνοδεύεται.

Ένα από τα προϊόντα μερικών σημαντικών ιστορικών προσπαθειών για να παρασχεθούν οι ακριβείς χαρακτηρισμοί της συμπεριφοράς των σημασιολογικών δικτύων και των frames, ήταν μια συναρτησιακή προσέγγιση στην αναπαράσταση γνώσης [Levesque, 1984]. Η ιδέα ήταν να δοθεί μια ακριβής προδιαγραφή της λειτουργίας που παρέχεται από μια βάση γνώσεων και συγκεκριμένα, των συμπερασμάτων που εκτελέσθηκαν από τη γνώση ανεξάρτητης βάσεως οποιασδήποτε εφαρμογής. Στην πράξη, η λειτουργική περιγραφή ενός συστήματος συλλογισμού διευκρινίζεται παραγωγικά μέσω μιας αποκαλούμενης διεπαφής "Tell&Ask". Μια τέτοια διεπαφή διευκρινίζει τις διαδικασίες που επιτρέπουν την κατασκευή βάσεων γνώσεων (Tell operations) και τις διαδικασίες που μας επιτρέπουν να πάρουμε πληροφορίες από τη βάση γνώσεων (Ask operations). Παρακάτω θα υιοθετήσουμε αυτήν την άποψη για το χαρακτηρισμό και τον καθορισμό μιας βάσης γνώσεων DL και των παραγωγικών υπηρεσιών που παρέχει.

Μέσα σε μια βάση γνώσεων μπορεί να υπάρξει μια σαφής διάκριση μεταξύ της εσωτερικής γνώσης (intensional knowledge) ή της γενικής γνώσης για το πεδίο ορισμού του προβλήματος, και τη επιπρόσθετης γνώσης (extensional knowledge), η οποία είναι συγκεκριμένη για ένα ιδιαίτερο πρόβλημα. Μια βάση γνώσεων DL είναι ανάλογα αποτελούμενη από δύο συστατικά – ένα "TBox" και ένα "ABox". Το TBox (Σώμα Όρων) περιέχει την εσωτερική γνώση υπό μορφή ορολογίας (ως εκ τούτου ο όρος "TBox" αλλά το "taxonomy" μπορεί να χρησιμοποιηθεί επίσης) και χτίζεται μέσω των δηλώσεων που περιγράφουν τις γενικές ιδιότητες των εννοιών. Λόγω της φύσης των σχέσεων υπαγωγής μεταξύ των εννοιών που αποτελούν την ορολογία, τα TBoxes θεωρούνται συνήθως σαν έχοντα μια μορφή πλέγματος (lattice-like structure). Αυτή η μαθηματική δομή κληρονομείται από την σχέση υπαγωγής – που ωστόσο δεν έχει καμία σχέση με οποιαδήποτε υλοποίηση. Το ABox (Σώμα Υποθέσεων) περιέχει επιπρόσθετη γνώση

(extensional) που καλείται επίσης και assertional γνώση (ως εκ τούτου "Abox") που είναι συγκεκριμένη για τα άτομα της περιοχής ενδιαφέροντος. Η εσωτερική γνώση θεωρείται συνήθως σαν μόνιμη, δηλαδη σαν γνώση που δεν αλλάζει με τον καιρό, και η επιπρόσθετη γνώση είναι ετεροκαθοριζόμενη ή εξαρτώμενη από ένα ενιαίο σύνολο περιστάσεων, και επομένως υπαγόμενη στην περιστασιακή ή ακόμα και σταθερή αλλαγή. Στο υπόλοιπο του τμήματος παρουσιάζουμε μια βασική διεπαφή Tell&Ask με την ανάλυση του TBox και του ABox μιας βάσης γνώσεων DL.

1.4.1 Το TBox (Σώμα Όρων)

Ένα βασικό στοιχείο μιας βάσης γνώσεων DL δίνεται από τις διαδικασίες που χρησιμοποιούνται για να χτίσουν την ορολογία. Τέτοιες διαδικασίες συσχετίζονται άμεσα με τις μορφές και την σημασία των δηλώσεων που επιτρέπονται στο TBox.

Η βασική μορφή δήλωσης σε ένα TBox είναι ο ορισμός έννοιας, δηλαδή ο καθορισμός μιας νέας έννοιας με βάση άλλες εκ των προτέρων ορισμένες έννοιες. Παραδείγματος χάριν, μια γυναίκα μπορεί να οριστεί ως ένα θηλυκό πρόσωπο με την παρακάτω δήλωση:

$$Γυναικα \equiv Προσωπο \sqcap Θηλυκο$$

Μια τέτοια δήλωση ερμηνεύεται συνήθως ως λογική ισοδυναμία, η οποία αντιστοιχεί σε παροχή των ικανών και αναγκαίων όρων για να χαρακτηριστεί ένα άτομο ως γυναίκα. Αυτή η μορφή ορισμού είναι πολύ ισχυρότερη από αυτή που χρησιμοποιείται σε άλλα είδη αναπαραστάσεων γνώσης, τα οποία επιβάλλουν μόνο τις αναγκαίες συνθήκες. Η δύναμη αυτού του είδους δήλωσης θεωρείται το χαρακτηριστικό γνώρισμα των βάσεων γνώσεων DL. Στις βάσεις γνώσεων DL, επομένως, μια ορολογία αποτελείται από ένα σύνολο ορισμών της ανωτέρω μορφής. Εντούτοις, υπάρχουν μερικές σημαντικές κοινές υποθέσεις που γίνονται συνήθως για τις ορολογίες DL:

- μόνο ένας ορισμός για ένα όνομα έννοιας επιτρέπεται και
- οι ορισμοί είναι ακυκλικοί, δηλαδή οι έννοιες ούτε καθορίζονται από μόνες τους (αυτοκαθορισμός) ούτε από άλλες έννοιες που αναφέρονται έμμεσα σε αυτές (ετεροκαθορισμός).

Αυτό το είδος περιορισμού είναι κοινό για πολλές βάσεις γνώσεων DL και υπονοεί ότι κάθε ορισμένη έννοια μπορεί να επεκταθεί με έναν μοναδικό τρόπο σε μια σύνθετη έκφραση, η οποία περιέχει μόνο τις ατομικές έννοιες εκείνες που προκύπτουν με την αντικατάσταση κάθε έννοιας με τη δεξιά πλευρά του ορισμού της.

Ο Nebel [1990b] έδειξε ότι ακόμη και η απλή επέκταση των ορισμών όπως αυτή, προκαλεί μια αναπόφευκτη πηγή πολυπλοκότητας. Στην πράξη εντούτοις, οι ορισμοί που αυξάνουν υπέρμετρα την πολυπλοκότητα του συλλογισμού δεν φαίνονται να εμφανίζονται. Σε αυτές τις περιπτώσεις η υπολογιστική πολυπλοκότητα των συμπερασμάτων μπορεί να μελετηθεί με τον τεμαχισμό της ορολογίας και με την εξέταση όλων των δεδομένων εννοιών ως πλήρως επεκταθείσες εκφράσεις. Επομένως, ένα μεγάλο μέρος της μελέτης των μεθόδων εξαγωγής συμπεράσματος στην Περιγραφική Λογική έχει στραφεί στις εκφράσεις εννοιών και πιό συγκεκριμένα, όπως συζητείται στο προηγούμενο τμήμα, στην υπαγωγή εννοιών σε άλλες (Subsumption), η οποία μπορεί να θεωρηθεί η βασική συλλογιστική υπηρεσία που παρέχεται στο TBox.

Ειδικότερα, ο βασικός στόχος στην κατασκευή μιας ορολογίας είναι η ταξινόμηση (classification), η οποία ισοδυναμεί σε τοποθέτηση μιας νέας έννοιας στην κατάλληλη θέση σε μια ταξονομική ιεραρχία των εννοιών. Η ταξινόμηση μπορεί να ολοκληρωθεί με την επαλήθευση της σχέσης υπαγωγής μεταξύ κάθε ορισμένης έννοιας στην ιεραρχία και της νέας – προς τοποθέτηση – έννοιας. Η τοποθέτηση της έννοιας θα είναι μεταξύ των πιό συγκεκριμένων εννοιών που υπάγονται στη νέα έννοια και στις γενικότερες έννοιες στις οποίες η νέα έννοια υπάγεται.

Οι γενικότερες διαδικασίες ορισμού εννοιών έχουν τύχει πρόσφατα ιδιαίτερης προσοχής, γεγονός που προέρχεται από τις προσπάθειες να καθιερωθούν οι επίσημες σχέσεις μεταξύ της Περιγραφικής Λογικής και των άλλων φορμαλισμών και από την επιδίωξη να ικανοποιηθεί η ανάγκη για την αυξανόμενη εκφραστική δύναμη. Ειδικότερα, η αποδοχή των κυκλικών ορισμών έχει οδηγήσει σε διαφορετικές σημασιολογικές ερμηνείες των δηλώσεων, γνωστές ως greatest/least fixed-point (μέγιστου/ελάχιστου σταθερού σημείου) και στην περιγραφική σημασιολογία. Αν και έχει υποστηριχτεί ότι διαφορετική σημασιολογία μπορεί να υιοθετηθεί ανάλογα με την εφαρμογή και το πεδίο ενδιαφέροντός της, συχνότερα υιοθετείται η περιγραφική σημασιολογία, η οποία απλά απαιτεί όλες οι δηλώσεις να ικανοποιούνται στην μετάφραση (interpretation). Επιπλέον, με τη κατάριψη της απαίτησης ότι στην αριστερή πλευρά ενός ορισμού μπορεί μόνο να υπάρξει ένα ατομικό όνομα έννοιας, μπορούμε να εξετάσουμε τα γενικά αξιώματα συνυπολογισμού (general inclusion axioms), της μορφής :

$$C \sqsubseteq D$$

όπου το C και το D είναι αυθαίρετες έννοιες. Είναι πρόδηλο δε, ότι ένας ορισμός έννοιας μπορεί να εκφραστεί από δύο γενικούς συνυπολογισμούς. Ως αποτέλεσμα διάφορων θεωρητικών μελετών σχετικά με την αποκρισιμότητα (decidability) και την

υλοποίηση τεχνικών για κυκλικά TBoxes, τα πιό πρόσφατα συστήματα DL επιτρέπουν ιδιαίτερα ισχυρούς κατασκευστές για τον ορισμό των εννοιών.

Η βασική υπηρεσία αφαίρεσης για τέτοια TBoxes μπορεί να αντιμετωπισθεί ως λογική επίπτωση και ανέρχεται σε επαλήθευση του εάν μια γενική σχέση (παραδείγματος χάριν μια σχέση υπαγωγής μεταξύ δύο εννοιών) είναι λογική συνέπεια των δηλώσεων στο TBox.

1.4.2 Το ABox (Σώμα Υποθέσεων ή Σώμα Ισχυρισμών)

Το ABox περιέχει την επιπρόσθετη γνώση (extensional knowledge) για την περιοχή ενδιαφέροντος, δηλαδή ισχυρισμούς για τα άτομα, αποκαλούμενα συνήθως ισχυρισμούς ιδιότητας μέλους (membership assertions). Παραδείγματος χάριν,

$$Θηλυκο \sqcap Προσωπο(ANNA)$$

δηλώνει ότι το άτομο ΑΝΝΑ είναι θηλυκό πρόσωπο. Λαμβάνοντας υπόψη τον ανωτέρω καθορισμό της γυναίκας, μπορεί να προκύψει από αυτόν τον ισχυρισμό ότι ΑΝΝΑ είναι μια περίπτωση της έννοιας γυναίκας. Ομοίως το,

εχειΠαιδι(ΑΝΝΑ ,JACOPO)

διευκρινίζει ότι η ΑΝΝΑ έχει τον JACOPO ως παιδί. Οι ισχυρισμοί του πρώτου είδους καλούνται επίσης ισχυρισμοί έννοιας, ενώ οι ισχυρισμοί του δεύτερου είδους καλούνται ισχυρισμοί ρόλου.

Όπως εμφανίζεται σε αυτά τα παραδείγματα, στο ABox η γνώση παριστάνεται υπό μορφή ισχυρισμών έννοιας και ισχυρισμών ρόλου. Στους ισχυρισμούς έννοιας οι γενικές εκφράσεις έννοιας επιτρέπονται, ενώ στους ισχυρισμούς ρόλου, όπου ο ρόλος δεν είναι ένας πρωτόγονος ρόλος αλλά μια έκφραση ρόλου, οι γενικές εκφράσεις έννοιας δεν επιτρέπονται, γεγονός που αντιμετωπίζεται στην περίπτωση των πολύ εκφραστικών γλωσσών μόνο.

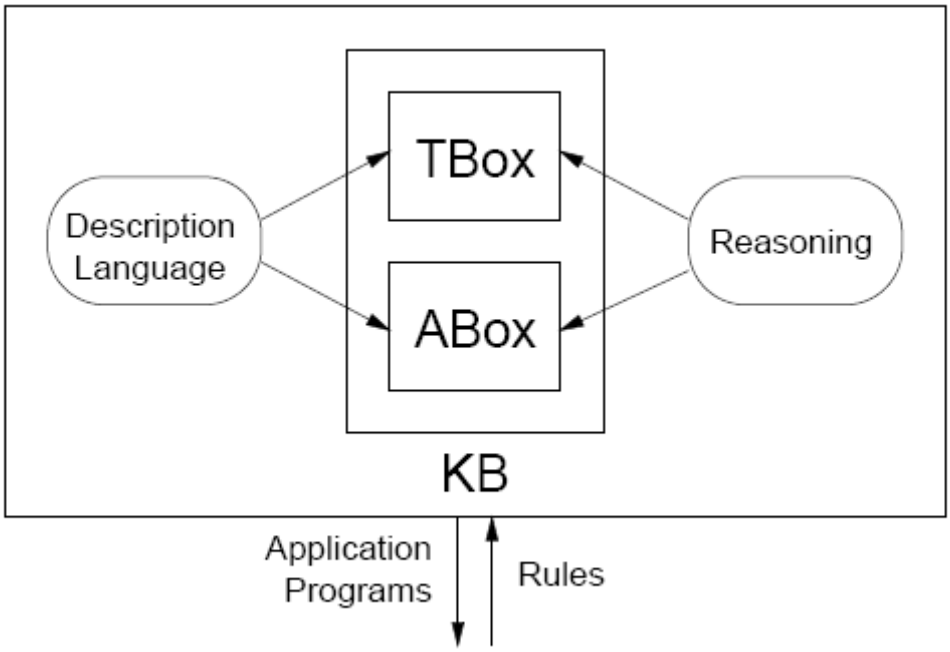
Ο βασικός στόχος συμπερασματολογίας σε ένα ABox είναι ο έλεγχος περίπτωσης (instance checking), ο οποίος ελέγχει εάν ένα δεδομένο άτομο είναι μια περίπτωση (ανήκει - *περιπίπτει*) σε μια καθορισμένη έννοια. Αν και συχνά χρησιμοποιούνται και άλλες συλλογιστικές μέθοδοι -υπηρεσίες, οι περισσότερες από αυτές καταλήγουν τελικά στον έλεγχο περίπτωσης. Μεταξύ τους βρίσκουμε τη συνέπεια βάσεων γνώσεων (knowledge base consistency), η οποία ισοδυναμεί με επαλήθευση του εάν κάθε έννοια

στη βάση γνώσεων αναγνωρίζει (υιοθετεί) τουλάχιστον ένα άτομο. Επίσης βρίσκουμε την πραγματοποίηση (realization), η οποία βρίσκει την πιό συγκεκριμένη έννοια της οποίας ένα μεμονωμένο αντικείμενο αποτελεί μια περίπτωση. Τέλος έχουμε την ανάκτηση, η οποία βρίσκει τα άτομα στη βάση γνώσεων που είναι περιπτώσεις μιας δεδομένης έννοιας. Όλες οι παραπάνω μέθοδοι μπορούν να επιτευχθούν με τη βοήθεια του ελέγχου περίπτωσης-στιγμιοτύπου.

Η παρουσία ατόμων σε μια βάση γνώσεων καθιστά το συλλογισμό πιό σύνθετο υπολογιστικά [Donini et Al, 1994b], και μπορεί να απαιτήσει σημαντικές επεκτάσεις μερικών τεχνικών εξαγωγής συμπεράσματος του TBox.

Αξίζει να τονιστεί ότι, αν και έχουμε διαχωρίσει για ευκολία τις υπηρεσίες για το ABox - όταν δεν μπορεί το TBox να εξεταστεί με τη βοήθεια του απλού μηχανισμού αντικατάστασης που χρησιμοποιείται για ακυκλικό TBoxes - οι συλλογιστικές υπηρεσίες μπορεί να χρειαστεί να λάβουν υπόψη όλη την βάση γνώσεων συμπεριλαμβανομένου του TBox και του ABox, και έτσι τα αντίστοιχα προβλήματα συλλογισμού να γίνουν πιό σύνθετα.

Παράλληλα πιό γενικές γλώσσες για τον καθορισμό ABoxes έχουν εξεταστεί. Τα συστήματα αναπαράστασης γνώσης που παρέχουν μια ισχυρή λογική γλώσσα για το ABox και μια γλώσσα DL για το TBox θεωρούνται συχνά *υβριδικά* συστήματα συλλογισμού, δεδομένου ότι οι απολύτως διαφορετικές γλώσσες αναπαράστασης γνώσης μπορούν να χρησιμοποιηθούν για να καθορίσουν τη γνώση στα διαφορετικά συστατικά. Τα υβριδικά συστήματα συλλογισμού ήταν δημοφιλή στη δεκαετία του '80 [Brachman et Al, 1985]. Πρόσφατα, το θέμα έχει επανακτήσει την προσοχή [Levy και Rousset, 1997 Donini et Al, 1998b], εστιάζοντας στις βάσεις γνώσεων που αποτελούνται από ένα συστατικό DL για τους ορισμούς εννοιών και ένα ακόμα συστατικό τμήμα λογικού προγραμματισμού για τους ισχυρισμούς (assertions) για τα άτομα. Οι έγκυρες και πλήρεις μέθοδοι εξαγωγής συμπεράσματος για τις υβριδικές βάσεις γνώσεων γίνονται δύσκολο να επινοηθούν στις περιπτώσεις που υπάρχει μια στενή αλληλεπίδραση μεταξύ των τμημάτων γνώσης.



Εικόνα 1.3

1.5 Γλώσσες Περιγραφής

Οι στοιχειώδεις γλωσσικές περιγραφές είναι οι ατομικές έννοιες και οι ατομικοί ρόλοι. Πιό σύνθετες περιγραφές μπορούν να χτιστούν από αυτούς επαγωγικά με τους κατασκευαστές εννοιών (concept constructors). Σε αφηρημένη γραφή, χρησιμοποιούμε τα γράμματα A και B για τις ατομικές έννοιες, το γράμμα R για τους ατομικούς ρόλους, και τα γράμματα C και D για τις περιγραφές έννοιας. Οι γλώσσες περιγραφής διακρίνονται από τους κατασκευαστές που παρέχουν. Στη συνέχεια θα συζητήσουμε τις διάφορες γλώσσες από την οικογένεια των *AL*-γλωσσών. Η γλώσσα *AL* (attributive language) εισήχθη από τους Schmidt - Schauss και Smolka [1991] ως μια ελάχιστη γλώσσα που χρήζει πρακτικού ενδιαφέροντος. Οι άλλες γλώσσες αυτής της οικογένειας είναι επεκτάσεις της *AL*.

Οι περιγραφές εννοιών στην AL διαμορφώνονται σύμφωνα με τον ακόλουθο κανόνα σύνταξης:

$$
\begin{aligned}
C, D \rightarrow\ & A \mid && (atomic\ concept)\\
& \top \mid && (universal\ concept)\\
& \bot \mid && (bottom\ concept)\\
& \neg A \mid && (atomic\ negation)\\
& C \sqcap D \mid && (intersection)\\
& \forall R.C \mid && (value\ restriction)\\
& \exists R.\top && (limited\ existential\ quantification)
\end{aligned}
$$

Αξίζει να σημειωθεί εδώ ότι στην *AL,* η άρνηση (negation) μπορεί να εφαρμοστεί μόνο στις ατομικές έννοιες, και μόνο η κορυφαία έννοια (υψηλότερη στην ιεραρχία) επιτρέπεται να χρησιμοποιείται στο πεδίο ενός υπαρξιακού ποσοδείκτη (existential quantification) για έναν ρόλο. Για ιστορικούς λόγους, η υπογλώσσα της *AL* που λαμβάνεται με την απαγόρευση της άρνησης για τα άτομα καλείται $\mathcal{FL}^{-}$ και η υπογλώσσα που λαμβάνεται από την $\mathcal{FL}^{-}$ με την απαγόρευση του περιορισμένου υπαρξιακού ποσοδείκτη καλείται $\mathcal{FL}_0$.

Προκειμένου να δωθεί μια αίσθηση του τι μπορεί να εκφραστεί στην *AL*, υποθέτουμε ότι το πρόσωπο και το θηλυκό είναι ατομικές έννοιες. Κατόπιν το $Προσωπο \sqcap Θηλυκο$ και $Προσωπο \sqcap \neg Θηλυκο$ είναι *AL*-έννοιες που περιγράφουν διαισθητικά, εκείνα τα πρόσωπα που είναι θηλυκά, και εκείνα που δεν είναι θηλυκά. Εάν, επιπλέον, υποθέσουμε ότι το $εχειΠαιδι$ είναι ένας ατομικός ρόλος, μπορούμε να διαμορφώσουμε το $Προσωπο \sqcap \exists εχειΠαιδι.T$ και το $Προσωπο \sqcap \forall εχειΠαιδι.Θηλυκο$, που δείχνει εκείνα τα πρόσωπα που έχουν ένα παιδί, και εκείνα τα πρόσωπα των οποίων όλα τα παιδιά είναι θηλυκά αντιστοιχα. Χρησιμοποιώντας την κατώτατη έννοια ($\bot$), μπορούμε επίσης να περιγράψουμε εκείνα τα πρόσωπα χωρίς παιδιά από το $Προσωπο \sqcap \forall εχειΠαιδι.\bot$.

Προκειμένου να καθοριστεί μια επίσημη σημασιολογία των *AL* - γλωσσών, θεωρούμε τις *διερμηνείες I* (interpretations) που αποτελούνται από μη κενό σύνολο Δ^{I} (το πεδίο της διερμηνείας) και μια λειτουργία μετάφρασης, η οποία αντιστοιχεί σε κάθε ατομική έννοια Α ένα σύνολο $A^I \subseteq \Delta^I$ και σε κάθε ατομικό ρόλο *R* μια δυαδική σχέση $R^I \subseteq \Delta^I \times \Delta^{\mathrm{I}}$. Η λειτουργία ερμηνείας επεκτείνεται στις περιγραφές εννοιών από τους ακόλουθους επαγωγικούς ορισμούς:

$$\top^{\mathcal{I}} = \Delta^I$$

$$\bot^I = \varnothing$$

$$(\neg A)^I = \Delta^I \setminus A^I$$

$$(C \sqcap D)^I = C^I \cap D^I$$

$$(\forall R.C)^I = \{a \in \Delta^I \mid \forall b.(a,b) \in R^I \rightarrow b \in C^I\}$$

$$(\exists R.\top)^I = \{a \in \Delta^I \mid \exists b.(a,b) \in R^I\}.$$

Λέμε ότι δύο έννοιες C, D είναι ισοδύναμες, και γράφουμε $C \equiv D$, εάν $C^I \equiv D^I$ για όλες τις διερμηνείες *I*.

1.5.1 Η οικογένεια των AL – γλωσσών

Είναι δυνατό να λάβουμε πιό εκφραστικές γλώσσες εάν προσθέτουμε περαιτέρω κατασκευαστές στο *AL*. Η ένωση των εννοιών (που υποδεικνύονται από το σύμβολο $\sqcup$) γράφεται ως $C \sqcup D$, και μεταφράζεται ως :

$$(C \sqcup D)^I = C^I \cup D^I$$

Πλήρης υπαρξιακή ποσοτικοποίηση (που υποδεικνύεται από την γράμμα *E*) γράφεται ως:

$$(\exists R.C)^I = \{a \in \Delta^I \mid \exists b.(a,b) \in R^I \wedge b \in C^I\}$$

Σημειώστε ότι $\exists R.C$ διαφέρει από το $\exists R.\top$ εξαιτίας του ότι και αυθαίρετες έννοιες επιτρέπονται να εμφανιστούν στο πεδίο του υπαρξιακού ποσοδείκτη.

Οι *Περιορισμοί αριθμού* (που υποδεικνύονται από το γράμμα *N*) που γράφονται ως $\geq nR$ (κατ' ελάχιστον περιορισμός) και ως $\leq nR$ (κατά μέγιστο περιορισμός), όπου το *n* ανήκει στους φυσικούς αριθμούς. Ερμηνεύονται δε ως

$$(\geq nR)^I = \left\{a \in \Delta^I \mid \left|\{b \mid (a,b) \in R^I\}\right| \geq n\right\}$$

και

$$(\leq nR)^I = \left\{a \in \Delta^I \mid \left|\{b \mid (a,b) \in R^I\}\right| \leq n\right\}$$

αντίστοιχα, όπου το «$|.|$» δείχνει τον πληθικό αριθμό των στοιχείων ενός συνόλου. Από σημασιολογικής άποψης, η κωδικοποίηση των αριθμών στους περιορισμούς αριθμού είναι ασήμαντη. Εντούτοις, για την ανάλυση πολυπλοκότητας των συμπερασμάτων έχει

ιδιαίτερη σημασία εάν ένας αριθμός *n* αναπαρίσταται στη δυαδική (ή δεκαδικό) μορφη ή από μια σειρά του μήκους *n,* δεδομένου ότι η δυαδική (δεκαδική) μορφή αποτελεί μια συμπαγέστερη αναπαράσταση.

Η *άρνηση* των αυθαίρετων εννοιών (που υποδεικνύονται από το γράμμα C, από το συμπλήρωμα – complement) γράφεται ως $\neg C$, και ερμηνεύεται ως :

$$(\neg C)^I = \Delta^I \setminus C^I$$

Με κάποιους πρόσθετους κατασκευαστές, μπορούμε να περιγράψουμε εκείνα τα πρόσωπα που έχουν είτε όχι περισσότερο από ένα παιδί είτε τουλάχιστον τρία παιδιά, ένα από το οποίο είναι θηλυκό:

$$\Pi\rho οσωπο \sqcap \left(\leq 1 \varepsilon χει \Pi αιδι \sqcup \left(\geq 3 \varepsilon χει \Pi αιδι \sqcap \exists \varepsilon χει \Pi αιδι.\Theta ηλυκο\right)\right)$$

Η επέκταση των *AL* γλωσσών από οποιοδήποτε υποσύνολο των ανωτέρω κατασκευαστών παράγει μια ιδιαίτερη *AL* γλώσσα. Ονομάζουμε κάθε *AL* γλώσσα προσθέτοντας ένα γράμμα της μορφής :

$$\mathcal{AL}[\mathcal{U}][\mathcal{E}][\mathcal{N}][\mathcal{C}]$$

όπου κάθε γράμμα στο όνομα αντιπροσωπεύει την παρουσία του αντίστοιχου κατασκευαστή. Παραδείγματος χάριν, η $\mathcal{ALEN}$ είναι η επέκταση του $\mathcal{AL}$ από τους πλήρεις υπαρξιακούς ποσοδείκτες και τους περιορισμούς αριθμού.

Από σημασιολογική άποψη, όλες αυτές οι γλώσσες δεν είναι ευδιάκριτες, εντούτοις. Η σημασιολογία επιβάλλει τις ισοδυναμίες :

$$C \sqcup D \equiv \neg(\neg C \sqcap \neg D) \qquad \kappa\alpha\iota$$

$$\exists R.C \equiv \neg \forall R.\neg C.$$

Ως εκ τούτου, η ένωση και ο πλήρης υπαρξιακός ποσοδείκτης μπορούν να εκφραστούν χρησιμοποιώντας την άρνηση. Αντιθέτως, ο συνδυασμός ένωσης και πλήρους υπαρξιακού ποσοδείκτη μας δίνει την δυνατότητα να εκφράσουμε την άρνηση των εννοιών (μέσω της ισοδύναμης κανονικής μορφής άρνησής τους). Επομένως, υποθέτουμε χωρίς βλάβη της γενικότητας ότι η ένωση και ο πλήρης υπαρξιακός ποσοδείκτης είναι διαθέσιμοι σε κάθε γλώσσα που περιέχει την άρνηση, και αντίστροφα. Συνεπώς (εξαιτίας της προαναφερθείσας ισοδυναμίας), όλες οι *AL*-γλώσσες μπορούν να γραφούν χρησιμοποιώντας τα γράμματα *U, E, N* μόνο. Είναι εύκολο να φανεί ότι οι οκτώ γλώσσες που προκύπτουν με αυτόν τον τρόπο είναι πράγματι ανά δύο μη-ισοδύναμες.

1.5.2 Οι γλώσσες περιγραφής ως τμήματα της κατηγορηματικής λογικής

Η σημασιολογία των εννοιών προσδιορίζει τις γλώσσες περιγραφής ως τμήματα της πρωτοβάαθμιας κατηγορηματικής λογικής. Δεδομένου ότι μια μετάφραση I αντίστοιχα ορίζει σε κάθε ατομική έννοια και ρόλο μια μοναδιαία και δυαδική σχέση πάνω στο Δ^I, μπορούμε να δούμε τις ατομικές έννοιες και τους ρόλους ως μοναδιαία και δυαδικά κατηγορήματα. Κατόπιν, οποιαδήποτε έννοια C μπορεί να μεταφραστεί αποτελεσματικά σε έναν τύπο κατηγορηματικής λογικής $\phi_C(x)$ με μια ελεύθερη μεταβλητή x έτσι ώστε για κάθε μετάφραση I το σύνολο στοιχείων Δ^I που ικανοποιούν το $\phi_C(x)$ να είναι ακριβώς C^I. Μια ατομική έννοια A μεταφράζεται στον τύπο A(x). Οι κατασκευαστές τομή, ένωση, και άρνηση μεταφράζονται στις λογικές σύζευξη, διάζευξη και άρνηση, αντίστοιχα. Εάν το C είναι ήδη μεταφρασμένο ως $\phi_C(x)$ και το R είναι ένας ατομικός ρόλος, τότε ο περιορισμός αξίας και ο υπαρξιακός ποσοδείκτης εκφράζονται από τους τύπους :

$$\phi_{\exists R.C}(y) = \exists x.R(y,x) \wedge \phi_C(x)$$

$$\phi_{\forall R.C}(y) = \forall x.R(y,x) \rightarrow \phi_C(x)$$

όπου το y είναι μια νέα μεταβλητή. Οι περιορισμοί αριθμού εκφράζονται από τους τύπους:

$$\phi_{\geq nR}(x) = \exists y_1,\ldots,y_n.R(x,y_1) \wedge \ldots \wedge R(x,y_n) \wedge \bigwedge_{i<j} y_i \neq y_j$$

$$\phi_{\leq nR}(x) = \forall y_1,\ldots,y_{n+1}.R(x,y_1) \wedge \ldots \wedge R(x,y_{n+1}) \wedge \bigvee_{i<j} y_i \neq y_j$$

1.6 Ορολογίες (Terminologies)

Στο εδάφιο αυτό θα γίνει αναφορά των βασικών αξιωμάτων των ορολογιών, τα οποία καθορίζουν τον τρόπο με τον οποίο οι έννοιες και οι ρόλοι συσχετίζονται μεταξύ τους. Κατόπιν επιλέγουμε τους ορισμούς σαν συγκεκριμένα αξιώματα και προσδιορίζουμε τις ορολογίες ως σύνολα ορισμών στα οποία μπορούμε να εισαγάγουμε τις ατομικές έννοιες ως συντμήσεις (abbreviations) ή ονόματα (names) για τις σύνθετες έννοιες. Εάν οι ορισμοί σε μια ορολογία περιέχουν κύκλους, πρέπει να υιοθετήσουμε σημασιολογία σταθερού σημείου (fixpoint semantics) για να τους καταστήσουμε σαφείς – μη αμφίσημους.

1.6.1 Τα αξιώματα ορολογιών

Στη γενική περίπτωση, τα αξιώματα ορολογιών έχουν τη μορφή :

$C \sqsubseteq D \quad (R \sqsubseteq S)$

ή

$C \equiv D \quad (R \equiv D)$

όπου τα *C, D* είναι έννοιες (και τα *R, S* είναι ρόλοι). Τα αξιώματα του πρώτου είδους καλούνται συμπεριλήψεις (inclusions), ενώ τα αξιώματα του δεύτερου είδους καλούνται ισότητες (equalities).

Η σημασιολογία των αξιωμάτων ορίζεται όπως είναι αναμενόμενο με βάση όσα έχουν αναφερθεί. Μια μετάφραση *I* ικανοποιεί μια συμπερίληψη :

$C \sqsubseteq D$ αν $C^I \subseteq D^I$ και μια ισότητα

$C \sqsubseteq D$ αν $C^I = D^I$.

Εάν το *T* είναι ένα σύνολο αξιωμάτων, τότε η μετάφραση *I* ικανοποιεί το *T* αν και μόνο αν η *I* ικανοποιεί κάθε στοιχείο του *T*. Εάν το *I* ικανοποιεί ένα αξίωμα (αντίστοιχα ένα σύνολο αξιωμάτων), λέμε ότι είναι ένα μοντέλο (model) αυτού του αξιώματος (αντίστοιχα του συνόλου των αξιωμάτων). Δύο αξιώματα ή δύο σύνολα αξιωμάτων είναι ισοδύναμα εάν έχουν τα ίδια μοντέλα

1.6.2 Ορισμοί

Μια ισότητα της οποίας η αριστερή πλευρά είναι μια ατομική έννοια είναι ένας ορισμός. Οι ορισμοί χρησιμοποιούνται για να εισαγάγουν συμβολικά ονόματα για σύνθετες περιγραφές. Παραδείγματος χάριν, με το αξίωμα :

$$Μητερα \equiv Γυναικα \sqcap \exists εχειΠαιδι.Προσωπο$$

συνδέουμε την περιγραφή στη δεξιά πλευρά με το όνομα Μητέρα. Τα συμβολικά ονόματα μπορούν να χρησιμοποιηθούν ως συντμήσεις και σε ορισμούς άλλων περιγραφών. Εάν, παραδείγματος χάριν, έχουμε καθορίσει τον πατέρα ανάλογα με τη μητέρα, μπορούμε να καθορίσουμε το γονέα ως :

$$Γονεας \equiv Μητερα \sqcup Πατερας$$

Ένα σύνολο ορισμών πρέπει να είναι σαφές – μη αμφίσημο. Καλούμε ένα πεπερασμένο σύνολο ορισμών T ως ορολογία ή TBox εάν κανένα συμβολικό όνομα δεν καθορίζεται περισσότερο από μία φορά, δηλαδή εάν για κάθε ατομική έννοια Α υπάρχει το πολύ ένα αξίωμα στο T η του οποίου αριστερή πλευρά είναι Α. Η παρακάτω περιγραφή παρουσιάζει μια ορολογία με τις σχετικες με τις οικογενειακές σχέσεις έννοιες :

$$Γυναικα \equiv Θηλυκο \sqcap Προσωπο$$

$$Ανδρας \equiv Προσωπο \sqcap \neg Θηλυκο$$

$$Μητερα \equiv Γυναικα \sqcap \exists εχειΠαιδι.Προσωπο$$

$$Πατερας \equiv Ανδρας \sqcap \exists εχειΠαιδι.Προσωπο$$

$$Γιαγια \equiv Μητερα \sqcap \exists εχειΠαιδι.Γονεας$$

$$ΜητεραΜεΠολλαΠαιδια \equiv Μητερα \sqcap \geq 3 εχειΠαιδι$$

$$ΜητεραΧωρισΚορη \equiv Μητερα \sqcap \forall εχειΠαιδι.\neg Θηλυκο$$

$$Συζηγος \equiv Γυναικα \sqcap \exists εχειΣυζηγο.Ανδρας$$

Εικόνα 1. 4

Ας υποθέσουμε οτι το T είναι μια ορολογία. Διαιρούμε τις ατομικές έννοιες που εμφανίζονται στο T σε δύο σύνολα :

1. τα σύμβολα ονομάτων N_T που εμφανίζονται στην αριστερή πλευρά κάποιου αξιώματος και

2. τα σύμβολα βάσεων B_T που εμφανίζονται μόνο στη δεξιά πλευρά των αξιωμάτων.

Τα σύμβολα ονομάτων καλούνται συχνά ορισμένες έννοιες και τα σύμβολα βάσεων καλούνται πρωτόγονες έννοιες. Είναι λοιπόν λογικό η ορολογία να ορίζει τα σύμβολα ονομάτων ως συνάρτηση των συμβόλων βάσεων.

Μια μετάφραση βάσης για την *T* είναι μια μετάφραση που ερμηνεύει μόνο τα σύμβολα βάσεων. Έστω ότι η *J* είναι μια τέτοια μετάφραση βάσεων. Μια μετάφραση *I* που ερμηνεύει επίσης τα σύμβολα ονομάτων είναι μια επέκταση της *J* εάν έχει την ίδια περιοχή με την *J,* δηλαδή $\Delta^I = \Delta^J$, και εάν συμφωνεί με την *J* για τα σύμβολα βάσεων. Λέμε ότι η *T* είναι *ορίσιμη (definitorial)* εάν κάθε ερμηνεία βάσεων έχει ακριβώς μια επέκταση που είναι ένα μοντέλο της *T*. Με άλλα λόγια, εάν ξέρουμε τι αντιπροσωπεύουν τα σύμβολα βάσεων, και η *T* είναι *definitorial*, τότε η έννοια των συμβόλων ονομάτων καθορίζεται πλήρως. Προφανώς, εάν μια ορολογία είναι *definitorial*, τότε κάθε ισοδύναμη ορολογία είναι επίσης definitorial. Η ερώτηση εάν μια ορολογία είναι definitorial ή όχι συσχετίζεται με την ερώτηση εάν οι ορισμοί της είναι ή όχι κυκλικοί. Παραδείγματος χάριν, η ορολογία που αποτελείται από το παρακάτω μόνο αξίωμα :

$$Ανθρωπος' \equiv Ζωο \sqcap \forall εχειΓονεα.Ανθρωπος' \quad (1.\ 1)$$

περιέχει έναν κύκλο, ο οποίος σε αυτήν την ειδική περίπτωση είναι πολύ απλός. Γενικά, ορίζουμε τους κύκλους σε μια ορολογία *T* ως εξής. Θεωρούμε τα *A*, *B* ως ατομικές έννοιες που εμφανίζονται στην *T*. Λέμε ότι το *A χρησιμοποιεί άμεσα* το *B* στην *T* εάν το *B* εμφανίζεται στη δεξιά πλευρά του ορισμού του *A,* και λέμε *χρησιμοποιεί* την μεταβατική κλειστότητα της σχέσης *χρησιμοποιεί άμεσα*. Η *T* περιέχει έναν κύκλο αν και μόνο αν υπάρχει εκεί μια ατομική έννοια στην *T* που *χρησιμοποιείται*. Διαφορετικά, η *T* καλείται ακυκλική.

Οι μοναδικές επεκτάσεις δεν χρειάζονται να υπάρχουν εάν μια ορολογία περιέχει κύκλους. Εξετάστε, παραδείγματος χάριν, την ορολογία που περιέχει μόνο το αξίωμα (1. 1). Εδώ, ο Ανθρωπος΄ είναι ένα σύμβολο ονόματος και το Ζώο και το εχειΓονεα είναι σύμβολα βάσεως. Για μια μετάφραση όπου το εχειΓονεα συνδέει κάθε Ζώο με τους προγόνους του, πολλές επεκτάσεις είναι δυνατό να μεταφράσουν το Ανθρωπος΄ με έναν έναν τέτοιο τρόπο ώστε το αξίωμα να είναι ικανοποιήσιμο: το Ανθρωπος΄ μπορεί, μεταξύ άλλων, να μεταφραστεί ως το σύνολο όλων των ζώων, ή ως το σύνολο μερικών ειδών, ή ως οποιοδήποτε άλλο σύνολο ζώων με την ιδιότητα ότι για ζώο περιέχει επίσης και τους προγόνους του.

Αντίθετα, εάν μια ορολογία *T* είναι ακυκλική, τότε είναι και ορίσιμή (definitorial). Ο λόγος είναι ότι μπορούμε να επεκτείνουμε μέσω μιας επαναληπτικής διαδικασίας τους ορισμούς της *T* με την αντικατάσταση κάθε εμφάνισης ενός ονόματος στη δεξιά πλευρά ενός ορισμού με τις έννοιες που αντιπροσωπεύει. Δεδομένου ότι δεν υπάρχει κανένας κύκλος στο σύνολο ορισμών, οι διαδικασίες τελικά ολοκληρώνονται και καταλήγουμε σε μια ορολογία *T'* που αποτελείται απλώς από ορισμούς του τύπου $A \equiv C'$, όπου *C'* περιέχει μόνο σύμβολα βάσεως και κανένα σύμβολο ονόματος. Καλούμε το *T'* επέκταση του *T*. Σημειώστε ότι το μέγεθος της επέκτασης μπορεί να είναι εκθετικό σε σχέση με το μέγεθος της αρχικής ορολογίας [Nebel, 1990b]. Η οικογένεια TBox στην εικόνα 1.5 είναι ακυκλική. Επομένως, μπορούμε να υπολογίσουμε την επέκτασή της, η οποία παρουσιάζεται στην εικόνα 1.5.

$$Γυναικα \equiv Θηλυκο \sqcap Προσωπο$$

$$Ανδρας \equiv Προσωπο \sqcap \neg(Προσωπο \sqcap Θηλυκο)$$

$$Μητερα \equiv (Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΠαιδι.Προσωπο$$

$$Πατερας \equiv (Προσωπο \sqcap \neg(Προσωπο \sqcap Θηλυκο)) \sqcap \exists εχειΠαιδι.Προσωπο$$

$$Γονεας \equiv ((Προσωπο \sqcap \neg(Προσωπο \cap Θηλυκο)) \sqcap \exists εχειΠαιδι.Προσωπο) \sqcup ((Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΠαιδι.Προσωπο)$$

$$Γιαγια \equiv ((Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΠαιδι.Προσωπο) \sqcap \exists εχειΠαιδι.(((Προσωπο \sqcap \neg(Προσωπο \sqcap Θηλυκο)) \sqcap \exists εχειΠαιδι.Προσωπο) \sqcup ((Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΠαιδι.Προσωπο))$$

$$ΜητεραΜεΠολλαΠαιδια \equiv ((Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΠαιδι.Προσωπο \sqcap \geq 3 εχειΠαιδι$$

$$ΜητεραΧωρισΚορη \equiv ((Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΠαιδι.Προσωπο) \sqcap \forall εχειΠαιδι.(\neg(Προσωπο \sqcap Θηλυκο))$$

$$Συζηγος \equiv (Προσωπο \sqcap Θηλυκο) \sqcap \exists εχειΣυζηγο.(Προσωπο \sqcap \neg(Προσωπο \sqcap Θηλυκο))$$

Εικόνα 1. 5

ΚΕΦΑΛΑΙΟ 2^{ο} : SEMANTIC WEB

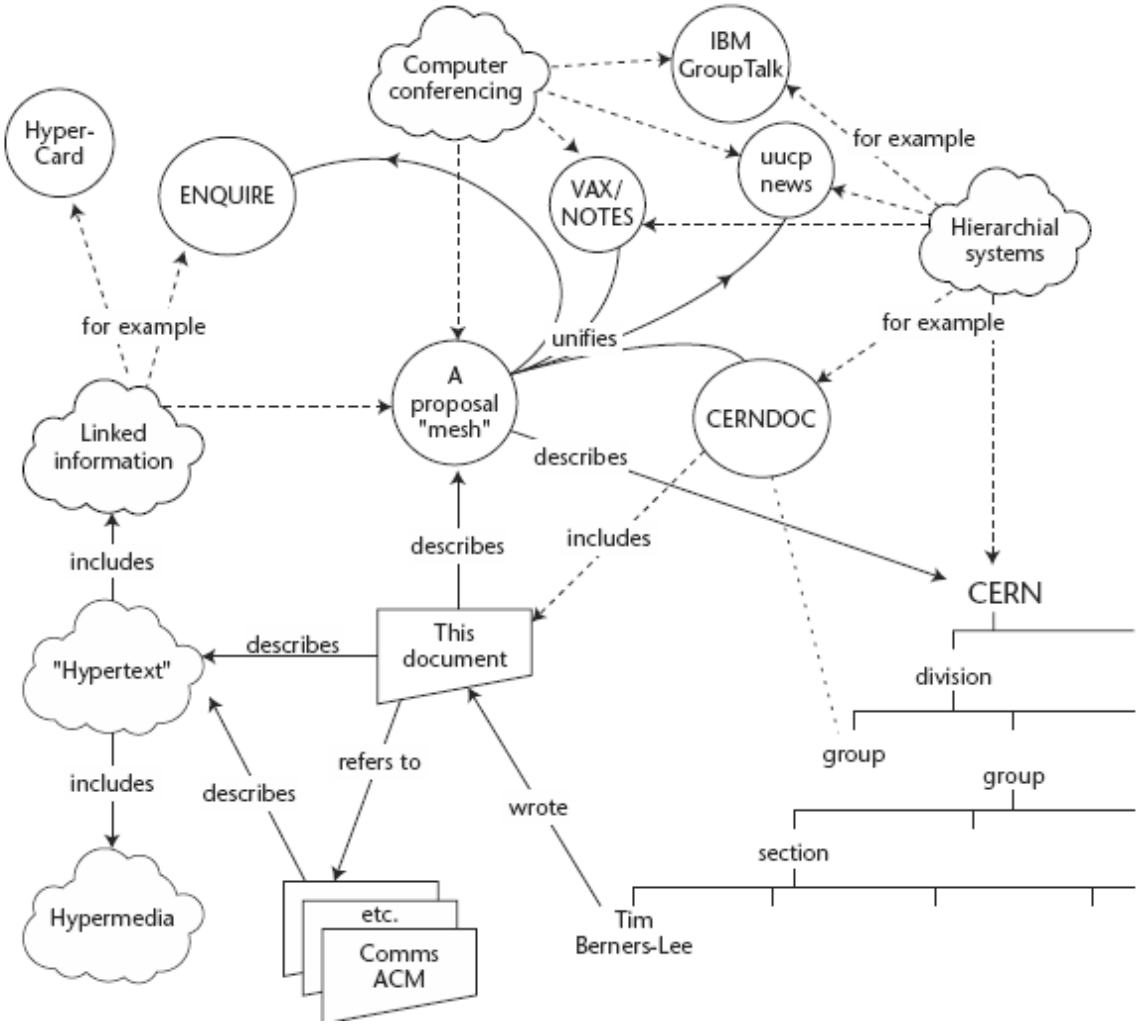
«Το πρώτο βήμα είναι να δομηθούν τα δεδομένα στο Διαδίκτυο σε μια μορφή που οι μηχανές να μπορούν φυσικά να τα καταλάβουν ή να τα μετατρέψουν σε αυτήν. Αυτή είναι η απαρχή του τι αποκαλώ Semantic Web, ένα δίκτυο δεδομένων που μπορούν να επεξεργαστούν άμεσα ή έμμεσα οι μηχανές»

Tim Berners Lee, "Weaving the Web", Harper San Francisco, 1999

2.1 Τι είναι το Semantic Web;

Ο Tim Berners Lee έχει ένα δισυπόστατο όραμα για το μέλλον του Διαδικτύου. Το πρώτο μέρος είναι να γίνει το Διαδίκτυο ένα πιό «συνεργάσιμο» μέσο. Το δεύτερο μέρος είναι να κατασταθεί το Διαδίκτυο κατανοητό, και έτσι επεξεργάσιμο, από τις μηχανές. Το σχήμα 2. 1 είναι αρχικό διάγραμμα του οράματός του Tim Berners Lee.

Το αρχικό όραμά του περιλάμβανε σαφώς κάτι περισσότερο από την ανάκτηση HTML σελίδων από τους υπολογιστές που αποτελούν το Διαδίκτυο. Στο σχήμα 2.1 βλέπουμε σχέσεις μεταξύ των στοιχείων πληροφοριών όπως "includes", "describes" και "wrote". Δυστυχώς, τέτοιες σχέσεις μεταξύ των πόρων του Διαδικτύου δεν συμπεριλαμβάνονται στην παρούσα δομή του Ιστού. Η τεχνολογία που δύναται να συλλάβει τέτοιες σχέσεις καλείται Resource Description Framework (RDF). Το βασικό σημείο που πρέπει να κατανοηθεί για το σχήμα 2.1 είναι ότι το αρχικό όραμα για το Διαδίκτυο κάλυπτε και πρόσθετα μετα-στοιχεία (meta-data) πέρα και πίσω από αυτό που θεωρούμε σήμερα το World Wide Web. Αυτά τα πρόσθετα μετα-στοιχεία απαιτείται οι μηχανές να είναι σε θέση να επεξεργαστούν, προκειμένου να λειτουργούν σε ένα επίπεδο σημασιολογικής και νοήμονος λειτουργίας στο Διαδίκτυο.



Σχήμα 2. 1

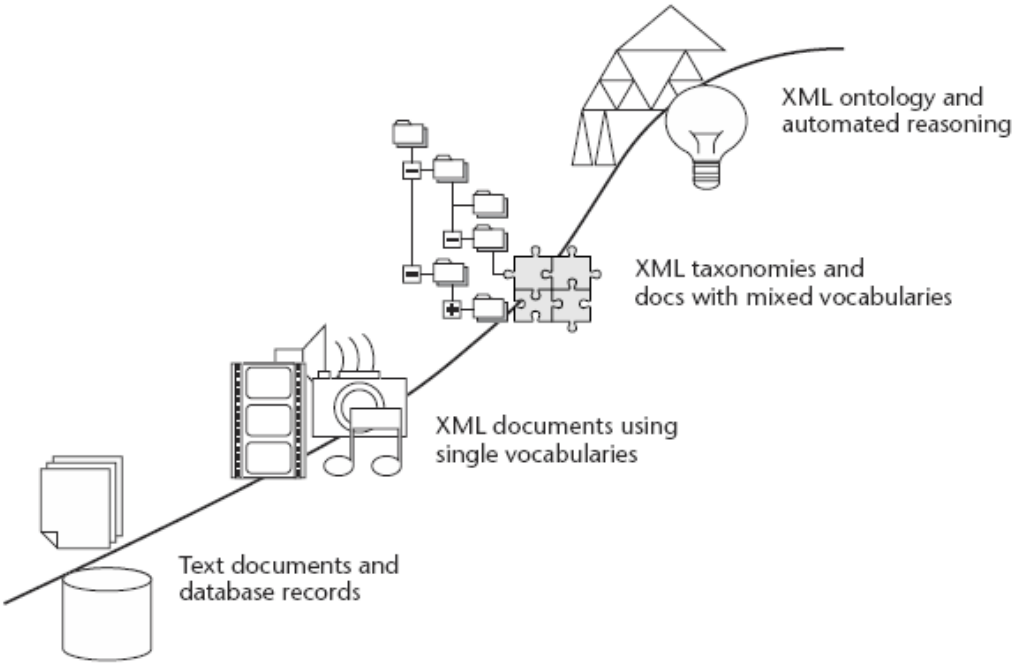
Το ερώτημα που σαφώς ενσκύπτει, είναι πώς δημιουργούμε ένα δίκτυο δεδομένων που οι μηχανές να μπορούν να επεξεργαστούν; Το πρώτο βήμα είναι μια ριζική μετατόπιση στον τρόπο που αντιλαμβανόμαστε τα δεδομένα. Ιστορικά, τα δεδομένα έχουν κρατηθεί μακριά από τις εφαρμογές και τον τρόπο δόμησής τους. Τα δεδομένα αυτούσια θεωρούνταν κάτι άσχετο ως προς τη επεξεργασία τους. Αυτή η ανακριβής τοποθέτηση προκάλεσε την έκφραση «απορρίματα μέσα, απορρίματα έξω» (Garbage In Garbage Out) ή GIGO. Η GIGO αποκαλύπτει βασικά το σφάλμα στο αρχικό επιχείρημα, με την καθιέρωση της εξάρτησης μεταξύ της επεξεργασίας των δεδομένων και των δεδομένων καθ'αυτών. Με άλλα λόγια, το χρήσιμο και αποδοτικό λογισμικό εξαρτάται πλήρως από τα καλά δομημένα δεδομένα. Οι άνθρωποι της Πληροφορικής άρχισαν να συνειδητοποιούν ότι το δεδομένο είναι σημαντικό και πρέπει να ελεγχθεί και να προστατευθεί. Οι γλώσσες προγραμματισμού άρχισαν να αποκτούν τις

αντικειμενοστρεφείς προεκτάσεις που έκαναν εσωτερικά τα δεδομένα την πρωταρχική τους προτεραιότητα. Εντούτοις, αυτή η προσέγγιση των δεδομένων κρατήθηκε εσωτερική στις εφαρμογές, έτσι ώστε οι κατασκευαστές λογισμικού να μπορούν να τηρήσουν την ιδιοκτησία την οργάνωσης των δεδομένων για ανταγωνιστικούς λόγους. Με το Διαδίκτυο, την eXtensible Markup Language (XML), και τώρα το αναδυόμενο Semantic Web, η δύναμη των εφαρμογών μετατοπίζεται από τις εφαρμογές στα δεδομένα. Αυτό μας δίνει επίσης το κλειδί στην κατανόηση του Semantic Web. Η πορεία στα επεξεργάσιμα από τις μηχανές δεδομένα θα καταστήσει τα δεδομένα εξυπνώτερα. Όλες οι τεχνολογίες που τώρα ανακύπτουν είναι τα θεμέλια μιας συστηματικής προσέγγισης στη δημιουργία των «έξυπνων δεδομένων». Το σχήμα 2.2 επιδεικνύει την πρόοδο των δεδομένων κατά μήκος μιας συνέχεια αυξανόμενης νοημοσύνης.

Το σχήμα 2.2 παρουσιάζει τα τέσσερα στάδια της πορείας των έξυπνων δεδομένων. Είναι αναμενόμενο ότι στην πορεία αυτή θα υπάρξουν κάποια λιγότερο ευδιάκριτα στάδια, καθώς επίσης και κάποια υβριδικά μοντέλα δεδομένων που ενδεχόμενα να μην υιοθετηθούν. Τα τέσσερα στάδια του διαγράμματος καλύπτουν ένα εύρος δεδομένων από εκείνα που διαθέτουν ελάχιστα στοιχεία νοημοσύνης μέχρι αυτά μου ενσωματώνουν αρκετές σημασιολογικές πληροφορίες ώστε να επιτρέπουν στις μηχανές να εξάγουν συμπεράσματα για αυτά. Έτσι έχουμε :

- Κείμενο (text) και βάσεις δεδομένων (προ-xml). Το αρχικό στάδιο όπου τα δεδομένα είναι στοιχεία ιδιόκτητα σε μια εφαρμογή. Κατά συνέπεια η νοημοσύνη είναι στην εφαρμογή και όχι στα στοιχεία.
- Έγγραφα XML για ένα συγκεκριμένο πεδίο. Το στάδιο όπου το δεδομένο επιτυγχάνει την ανεξαρτησία από την εφαρμογή μέσα σε ένα συγκεκριμένο πεδίο. Τα δεδομένα είναι τώρα αρκετά έξυπνα ώστε να κινηθούν μεταξύ των εφαρμογών σε μια αυστηρά ωστόσο καθορισμένη περιοχή. Ένα παράδειγμα αυτού είναι τα πρότυπα XML στη βιομηχανία υγειονομικής περίθαλψης, την ασφαλιστική βιομηχανία, ή τη βιομηχανία ακίνητων περιουσιών.
- Ταξονομίες και έγγραφα με τα μικτά λεξιλόγια. Σε αυτό το στάδιο, τα δεδομένα μπορούν να συντεθούν από πολλαπλές περιοχές και να ταξινομηθούν σε μια ιεραρχική ταξονομία. Στην πραγματικότητα, η ταξινόμηση μπορεί να χρησιμοποιηθεί για την ανακάλυψη των δεδομένων. Οι απλές σχέσεις μεταξύ των κατηγοριών στην ταξονομία μπορούν να χρησιμοποιηθούν για να συνδέσουν και έτσι να συνδυάσουν τα δεδομένα.

Κατά συνέπεια, το δεδομένο είναι τώρα αρκετά έξυπνο ώστε να ανακαλυπτεται εύκολα και να συνδυάζεται λογικά με άλλα δεδομένα.



Σχήμα 2. 2

- Οντολογίες και κανόνες. Σε αυτό το στάδιο, τα νέα δεδομένα μπορούν να προκύψουν από τα υπάρχοντα δεδομένα με την ακολουθία των λογικών κανόνων. Στην ουσία, το δεδομένο είναι τώρα αρκετά έξυπνο ώστε να μπορεί να περιγραφεί με συγκεκριμένες σχέσεις και περίπλοκους φορμαλισμούς, όπου οι λογικοί υπολογισμοί μπορούν να γίνουν με την «σημασιολογική άλγεβρα». Αυτό επιτρέπει το συνδυασμό και τον επανασυνδυασμό των δεδομένων σε πιό ατομικό επίπεδο και την πολύ λεπτομερή ανάλυσή τους. Κατά συνέπεια, σε αυτό το στάδιο, το δεδομένο δεν υπάρχει πλέον ως σταγόνα αλλά ως μέρος ενός περίπλοκου μικρόκοσμου. Ένα παράδειγμα αυτής της εκλέπτυνσης στην δόμηση των δεδομένων είναι η αυτόματη «μετάφραση» ενός εγγράφου από μια περιοχή στο ισοδύναμο (ή όσο το δυνατόν πιό κοντά) έγγραφο σε μια άλλη περιοχή.

Μπορούμε τώρα να συνθέσουμε έναν νέο ορισμό του Semantic Web: ένας επεξεργαζόμενος από μηχανές Ιστός (machine-processable Web) έξυπνων δεδομένων. Επιπλέον, μπορούμε περαιτέρω να ορίσουμε τα έξυπνα δεδομένα ως στοιχεία που είναι ανεξάρτητα από την εκάστοτε εφαρμογή, ικανά να ανασυντεθούν, ταξινομημένα

ιεραρχικά και μέρος ενός μεγαλύτερου «οικοσυστήματος» πληροφοριών που αποκαλούμε οντολογία. Η κοινοπραξία World Wide Web (W3Consortium) έχει καθιερώσει μια δραστηριότητα (που αποτελείται από διάφορες ομάδες) που αφιερώνεται στην εφαρμογή του οράματος του Semantic Web.

2.2 Γιατί χρειαζόμαστε το Semantic Web;

Το Semantic Web δεν είναι μόνο για το World Wide Web. Αντιπροσωπεύει ένα σύνολο τεχνολογιών που θα λειτουργήσει εξίσου αποδοτικά και στα εσωτερικά εταιρικά intranets. Αυτό είναι ανάλογο με τις υπηρεσίες Διαδικτύου που αντιπροσωπεύουν τμήματα υπηρεσιών στο ενδοδίκτυο μιας εταιρίας. Έτσι, το Semantic Web θα επιλύσει διάφορα βασικά προβλήματα που αντιμετωπίζουν οι τρέχουσες αρχιτεκτονικές τεχνολογίες πληροφοριών.

2.2.1 Υπερφόρτωση Πληροφοριών (Information Overloading)

Η υπερφόρτωση πληροφοριών είναι το προφανέστερο πρόβλημα που χρήζει λύσης και οι τεχνολόγοι εμπειρογνώμονες έχουν προειδοποιήσει ήδη εδώ και 50 έτη. Στο άρθρο “Ovelcoming Information Overload” (Ξεπερνώντας την Υπερφόρτωση Πληροφοριών – InfoWorld, January 7, 2000), ο Paul Krill αναφέρει χαρακτηριστικά ότι, «αυτή η κατάσταση προκύπτει από την διατήρηση ενός γρήγορου ρυθμού ανάπτυξης του μεγέθους των πληροφοριών που είναι διαθέσιμες, ενώ οι ώρες της ημέρας παραμένουν 24 και οι εγκέφαλοί μας παραμένουν κατά προσέγγιση στην ίδια κατάσταση ανάπτυξης όπως ήταν τότε που οι άνθρωποι αντάλλασαν μηνύματα πάνω σε πέτρες». Είναι φυσικό, ότι αυτό το πρόβλημα έχει οπωσδήποτε διογκωθεί με τη διάδοση του Διαδικτύου, του ηλεκτρονικού ταχυδρομείου, και τώρα του στιγμιαίου μηνύματος. Δυστυχώς, η προκατάληψή μας υπέρ της παραγωγής και κατά της επαναχρησιμοποίησης της γνώσης έχει αφήσει αυτό το πρόβλημα εκκρεμές έως ότου έχει λάβει τελικά μια τραγική αναλογία.

Μια υπενθύμιση της αποτυχίας μας να σημειώσουμε πρόοδο σε αυτό το ζήτημα είναι και η προειδοποίηση του Vannevar Bush το 1945 όταν είπε: «Τώρα πιά δημιουργείται ένα βουνό ανάπτυξης στην έρευνα. Ωστόσο είναι ολοένα αυξανόμενη η πεποίθηση ότι αδύνατούμε να σκαρφαλώσουμε καθώς η εξειδίκευση επεκτείνεται. Ο ερευνητής τρικλίζεται από τις διαπιστώσεις και τα συμπεράσματα χιλιάδων άλλων

εργαζομένων - συμπεράσματα που δεν μπορεί να βρεί το χρόνο να αντιληφθεί και πολύ λιγότερο να τα θυμηθεί, όπως εμφανίζονται. Παρ'όλα αυτά η εξειδίκευση γίνεται όλο και πιο απαραίτητη για την πρόοδο, και η προσπάθεια που οφείλει να καταβάλει κανείς προκειμένου να γεφυρώσει το χάσμα μεταξύ των απόψεων είναι ακολούθως υπερφυσική».

2.2.2 Stovepipe Συστήματα

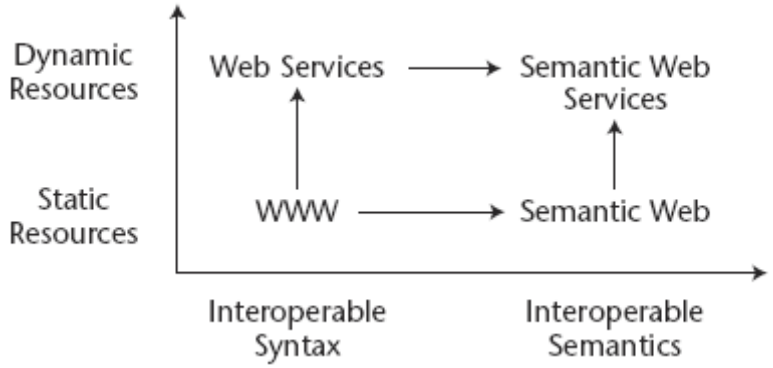
Ένα stovepipe σύστημα είναι ένα σύστημα όπου όλα τα συστατικά του είναι συνδεδεμένα με τέτοιο τρόπο ώστε να μπορούν να εργαστούν κατά ένα συγκεκριμένο τρόπο μόνο. Επομένως, οι πληροφορίες ρέουν μόνο στο stovepipe και δεν μπορούν να μοιραστούν σε άλλα συστήματα ή επιχειρήσεις που τα χρειάζονται. Παραδείγματος χάριν, ο client μπορεί μόνο να επικοινωνήσει με ένα συγκεκριμένο middleware που «αντιλαμβάνεται» και λειτουργεί αποκλειστικά με ένα δεδομένο σχήμα βάσης δεδομένων. Χαρακτηριστικό είναι το παράδειγμα των Kent Wreder και Yi Deng που περιγράφουν το πρόβλημα σε ένα σύστημα πληροφοριών υγειονομικής περίθαλψης:

«Στο παρελθόν, αυτά τα συστήματα χτίστηκαν βασισμένα στις εκάστοτε μεμονωμένες χρήσεις, ορμώμενα από μια αποσπασματική μόδα και συνδεόμενα στενά με κατά παραγγελία δημιουργημένα μέσα. Αυτή η πρακτική οδήγησε στα stovepipe συστήματα που έχουν πολλές επαναλαμβανόμενες λειτουργίες, είναι μονολιθικά, δεν επιτρέπουν την εξέλιξη και δεν είναι διαλειτουργικα (inter-operable). Το ζητούμενο για την βιομηχανία υγειονομικής περίθαλψης είναι να βρεθεί ο τρόπος που θα μπορέσει να μεταβεί μια επιχείρηση από αυτά τα stovepipe συστήματα στα ανοικτά συστήματα πληροφοριών υγειονομικής περίθαλψης επόμενης γενεάς που είναι διαλειτουργικά, επεκτάσιμα και συντηρήσιμα».

Η απόδραση από τα stovepipe συστήματα πρέπει να γίνει συντονισμένα και σε όλες τις τάξεις των επιχειρηματικών αρχιτεκτονικών πληροφορικής. Οι σημασιολογικές τεχνολογίες Ιστού θα είναι οι αποτελεσματικότερες στο επιχειρούμενο διαχωρισμό από τα μονολιθικά και παλαιωμένα συστήματα βάσεων δεδομένων.

Πρόσφατα, ο χειρωνακτικός συντονισμός βάσεων δεδομένων ήταν επιτυχής στην επίλυση της περίπτωσης του ελεύθερου σκοπευτή της Ουάσιγκτον. Ο Johnathan Alter του Newsweek περιγράφει την επιτυχία αυτή ως εξής: «Ήταν με το ταίριασμα ενός αποτυπώματος που βρέθηκε σε έναν κατάλογο πυροβόλων όπλων σε έναν τόπο εγκλήματος στο Μοντγκόμερυ και με τη χρήση μιας βάσης δεδομένων INS στην πολιτεία της Ουάσιγκτον. Τα στοιχεία αυτά χρησιμοποίησε η αστυνομία προκειμένου να εξιχνιάσει την περίπτωση και να προετοιμάσει το έδαφος για τη σύλληψη των δύο

πιθανών ελεύθερων σκοπευτών...Ακόμη περισσότερα στοιχεία ήταν διαθέσιμα, αλλά δεν ήταν συσχετισμένα μεταξύ τους έως ότου ήταν πάρα πολύ αργά, όπως για παράδειγμα τα αρχεία των παραβιάσεων του κώδικα οδικής κυκλοφορίας του ελεύθερου σκοπευτή στις πρώτες ημέρες της δράσης του».



2.2.3 Φτωχή συνάθροιση περιεχομένων (Poor Content Aggregation)

Το να συσχετιστούν πληροφορίες από ανόμοιες πηγές είναι ένα επαναλαμβανόμενο πρόβλημα σε διάφορα πεδία εργασίας και έρευνας όπως η συνάθροιση οικονομικού απολογισμού, η συνάθροιση πληροφοριών από δικτυακούς τόπους με ειδησεογραφικό περιεχόμενο, οι συγκριτικές αγορές και η εξόρυξη πληροφοριών από συστήματα αποθήκευσής τους. Δυστυχώς, η πιό κοινή τεχνική για αυτές τις δραστηριότητες είναι η χειρωνακτική και η κατά περίπτωση υλοποίηση. Ο Bill Orr περιγράφει την πρακτική ως εξής :

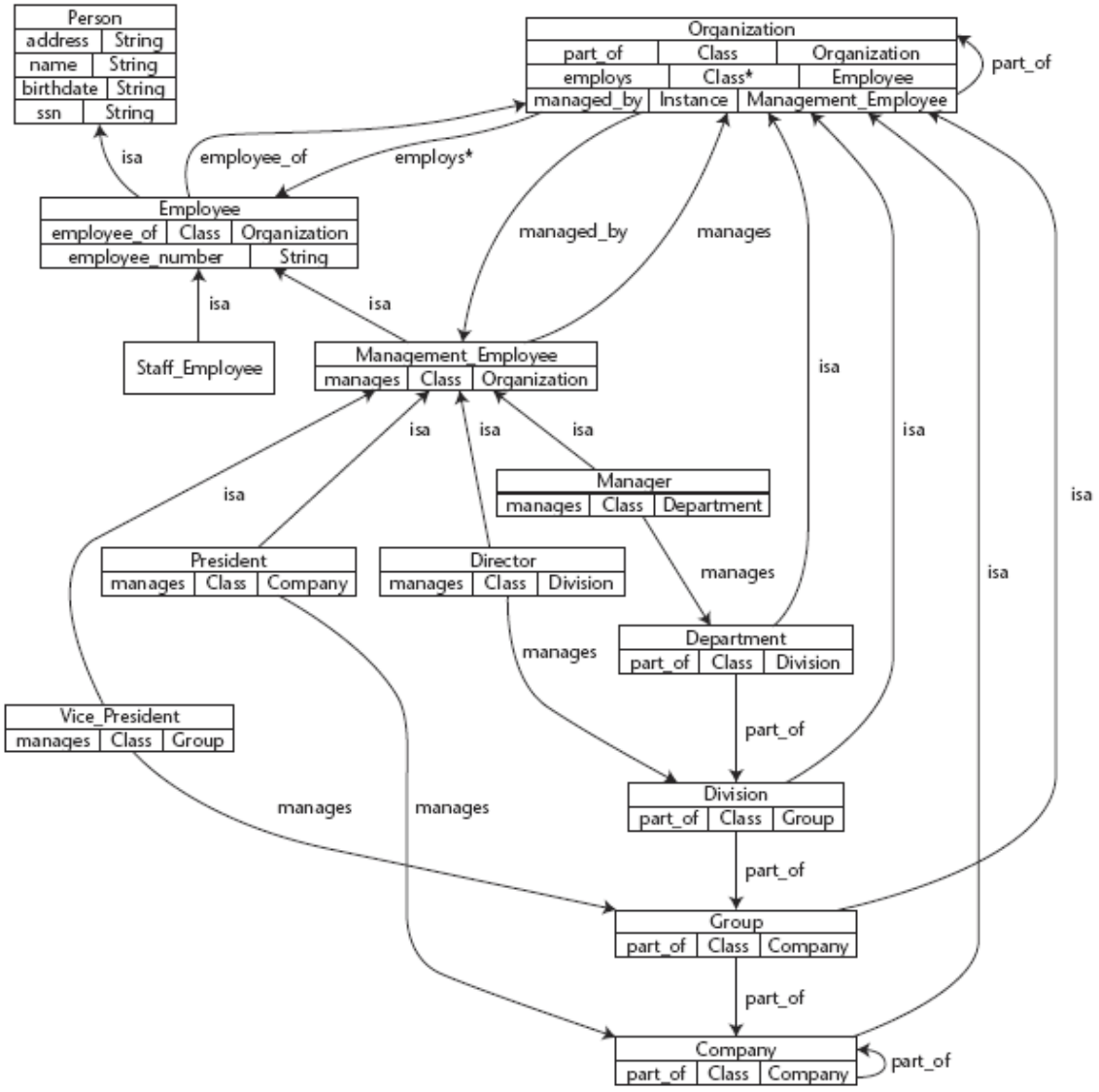
«Η τεχνολογία της συγκέντρωσης στοιχείων λογαριασμών δεν είναι επιστήμη εκτίναξης πυραύλων. Η πρωταρχική μέθοδος ξεκίνησε από το διακριτικό όνομα “screen scraping”. Το κύριο μειονέκτημα αυτής της μεθόδου είναι ότι αναζητά τα μηνύματα που γράφονται στην HTML, τα οποία περιγράφονται από το εκάστοτε σχήμα (μέγεθος τύπων, διάστημα παραγράφου, κλπ....) αλλά δεν δίνουν καμία σημασία για την έννοια ενός εγγράφου. Έτσι ο προγραμματιστής που δημιουργεί έναν νέο λογαριασμό και αναζητά να υπολογίσει την κατάσταση ισορροπίας του, πάντα εμφανίζεται σε μια ορισμένη θέση στην οθόνη. Το πρόβλημα έρχεται όταν προκύπτουν κάποιες αλλαγές θέσης ή ονόματος. Έτσι αυτή η μέθοδος απαιτεί πολύ και συνεχή συντήρηση».

2.3 Οντολογίες

Το σχήμα 2.4 παρουσιάζει μια απλή οντολογία του ανθρώπινου δυναμικού μιας επιχείρισης που δημιουργήθηκε με το εργαλείο οντολογιών Protege (http://protege.stanford.edu). Θα παρατηρήσετε ότι υπάρχουν κατηγορίες όπως το πρόσωπο, η οργάνωση και ο υπάλληλος. Σε μια οντολογία, αυτοί οι όροι καλούνται έννοιες, επειδή αντιστοιχούν στις διανοητικές έννοιες που τα ανθρώπινα οντα κατανοούν ως ένα ιδιαίτερο τμήμα μιας περιοχής γνώσης όπως αυτού του ανθρώπινου δυναμικού.

Αυτές οι έννοιες και οι σχέσεις που εφαρμόζονται μεταξύ τους συνήθως αναφέρονται ως κατηγορίες, σχέσεις, ιδιότητες, χαρακτηριστικά και τιμές (των ιδιοτήτων/ χαρακτηριστικών). Έτσι αυτό που το σχήμα 2.4 απεικονίζει πρώτιστα είναι έννοιες των σημαντικών οντοτήτων της περιοχής, οι οποίες υλοποιούνται ως κατηγορίες. Παραδείγματα αυτών είναι το πρόσωπο, η οργάνωση, και ο υπάλληλος. Επίσης απεικονίζονται οι σχέσεις μεταξύ αυτών των εννοιών, όπως το employee_of, managed_by, και manages. Τέλος, οι ιδιότητες και τα χαρακτηριστικά απεικονίζονται. Τα παραδείγματα περιλαμβάνουν τη διεύθυνση,το όνομα,την ημερομηνία γεννήσεως, και το ssn κάτω από την κατηγορία προσώπων. Αυτές οι ιδιότητες έχουν είτε ρητές τιμές είτε, συχνότερα, έχουν ένα εύρος τιμών. Το εύρος τιμών για τα χαρακτηριστικά της ιδιότητας του employee_of, μια ιδιότητα της έννοιας του υπαλλήλου είναι παραδείγματος χάριν η έννοια της οργάνωσης. Με το εύρος εννοούμε ότι οι μόνες πιθανές τιμές για οποιεσδήποτε περιπτώσεις της ιδιότητας employee_of που καθορίζεται για την έννοια του υπαλλήλου πρέπει να προέλθουν από την έννοια οργάνωση.

Βλέπουμε λοιπόν άμεσα ότι μια οντολογία «προσπαθεί» να συλλάβει την σημασία (αυτό που αποκαλούμε σημασιολογία) μιας ιδιαίτερης θεματικής περιοχής ή ενός τομέα γνώσης που αντιστοιχεί σε αυτό που ένας άνθρωπος ξέρει για εκείνη την περιοχή. Μια οντολογία χαρακτηρίζει επίσης την σημασία από την άποψη των εννοιών και των σχέσεών τους.



Σχήμα 2.4

Η λίστα 2.5 είναι ένα απόσπασμά της κειμενικής όψης της ίδιας οντολογίας. Στην περίπτωση της λίστας 2.5, η χρησιμοποιούμενη γλώσσα είναι η Open Knowledge Base Connectivity Language (OKBC, http://www. ai. sri. com/~okbc/). Το παράδειγμα αυτό υπογραμμίζει ένα σημαντικό σημείο: Δεν υπάρχει καμία λογική διαφορά μεταξύ μιας γραφικής και της ισοδύναμης κειμενικής απόδοσης μιας οντολογίας (ή οποιουδήποτε άλλουδήποτε προτύπου, για αυτό το θέμα). Αυτό το γεγονός είναι σημαντικό, επειδή ένα βασικό σημείο της όλης προσέγγισης είναι ότι μια οντολογία αντιπροσωπεύεται με μια γλώσσα αντιπροσώπευσης γνώσης (όπως μια σημασιολογική γλώσσα Ιστού όπως RDF/S, DAML+OIL, OWL, ή σε μια γλώσσα οντολογίας που προηγείται χρονικώς του

σημασιολογικού Ιστού, όπως η λογική Ontolingua/KIF/Common, OKBC, CycL, ή Prolog). Επιπλέον, τέτοιες γλώσσες οντολογίας είναι βασισμένες σε μια ιδιαίτερη λογική, με αυτή τη λογική να είναι μια γλώσσα με αυστηρά καθορισμένη σύνταξη και σημασιολογία. Μερικές φορές, επομένως, καλούμε τη γλώσσα στην οποία η οντολογία αντιπροσωπεύεται μια λογικά-βασισμένη γλώσσα. Το σημαντικό ζήτημα είναι αυτό της δύναμης της ελλοχεύουσας-υποννοούμενης γλώσσας που χρησιμοποιείται για να αντιπροσωπεύσει την οντολογία.

```
(defclass
     (is-a USER)
     (role concrete)
     (single-sot managed_by
          (type SYMBOL)
;+        (allowed-classes Management_Employee)
;+        (cardinality 1 1)
          (create-accessor read-write))
     (single-slot part_of
          (type SYMBOL)
;+        (allowed-parents Organization)
;+        (cardinality 0 1)
          (create-accessor read-write))
     (multislot employs
          (type SYMBOL)
;+        (allowed-parents Employee)
          (cardinality 1 ?VARIABLE)
          (create-accessor read-write)))

(defclass Department
     (is-a Organization)
     (role concrete)
     (single-slot part_of
          (type SYMBOL)
;+        (allowed-parents Division)
;+        (cardinality 0 1)
          (create-accessor read-write)))


(defclass Company
     (is-a Organization)
     (role concrete)
     (single-slot part_of
          (type SYMBOL)
;+        (allowed-parents Company)
;+        (cardinality 0 1)
          (create-accessor read-write)))

(defclass Person
```

```
      (is-a USER)
      (role concrete)
      (single-slot birthdate
           (type STRING)
;+         (cardinality 1 1)
           (create-accessor read-write))
      (single-slot name_
           (type STRING)
;+         (cardinality 1 1)
           (create-accessor read-write))
      (single-slot address
           (type STRING)
;+         (cardinality 1 1)
           (create-accessor read-write))
      (single-slot ssn
           (type STRING)
;+         (cardinality 1 1)
           (create-accessor read-write)))

... some classes omitted for brevity
```

Λίστα 2. 5

Ο ακρογωνιαίος λίθος της θεωρίας των οντολογιών είναι οι high-end γλώσσες οντολογιών που υποστηρίζονται από μια αυστηρά μαθηματικά θεμελιωμένη λογική, η οποία με αυτόν τον τρόπο κάνει την οντολογία ερμηνεύσιμη απ'ευθείας από τις μηχανές (machine-interpretable). Αυτό το μοναδικό χαρακτηριστικό τους που συνεπάγεται ότι το σημασιολογικό πρότυπο είναι ερμηνεύσιμο από τη μηχανή, καταδεικνύει ότι ο υπολογιστής και το λογισμικό του μπορούν να ερμηνεύσουν τη σημασιολογία του προτύπου άμεσα χωρίς καμία ανθρώπινη συμμετοχή. Το λογισμικό που υποστηρίζεται από τις οντολογίες κινείται μέχρι την ανθρώπινη γνώση, δηλαδή το εννοιολογικό επίπεδο και οι άνθρωποι δεν ειναι απαραίτητο να κινηθούν προς το επίπεδο των μηχανών. Η αλληλεπίδραση με τους υπολογιστές πραγματοποιείται στο δικό μας επίπεδό και όχι στο επίπεδο μηχανής. Αυτό είναι ένα εξαιρετικά σημαντικό σημείο, και υπογραμμίζει την αξία των οντολογιών.

2.3.1 Οι ορισμοί οντολογιών

Μια οντολογία καθορίζει τις κοινές λέξεις και τις έννοιες (την σημασία τους) που χρησιμοποιούνται για να περιγράψουν και να αναπαραστήσουν έναν τομέα της γνώσης.

Εάν αναζητήσουμε τον ορισμό της λέξης οντολογία στο λεξικό, θα βρούμε τον ακόλουθο: (από Merriam - Webster OnLine: http://www. m-w. com/):

- Ένας κλάδος της μεταφυσικής σχετικός με την φύση και τις σχέσεις της ύπαρξης
- Μια συγκεκριμένη θεωρία για τη φύση της ύπαρξης ή τα είδη των υπαρκτών οργανισμών.

Αυτός ο ορισμός δείχνει ότι ο όρος προέρχεται από την φιλοσοφία - συγκεκριμένα, ένα μέρος της μεταφυσικής που αφορά τη συστηματική μελέτη των αρχών που κρύβονται κάτω από ένα ιδιαίτερο θέμα, όπως για παράδειγμα τη φύση της ύπαρξης και τη φύση της εμπειρίας. Συχνά γίνεται η διάκριση μεταξύ οντολογίας «κεφαλαίου Ο» και της «μικρού ο». Η οντολογία «κεφαλαίου Ο» αντιπροσωπεύει την φιλοσοφική θεώρηση. Η οντολογία «μικρού ο» αντιπροσωπεύει την προσέγγιση την επιστήμης της Πληροφορικής που έχει προκύψει κατά τη διάρκεια των οκτώ προηγούμενων ετών. Η επιστήμη της Πληροφορικής παραθέτει προς τούτο κάποιους ορισμούς για την εν λόγω έννοια. Ο πρώτος ορισμός είναι μια απλή παράφραση σε καθημερινή γλώσσα του πιό τεχνικού δεύτερου ορισμού, αλλά για να καταλάβει κανείς το δεύτερο, είναι καλό να στηριχτεί σε μια διευκρίνιση του πρώτου:

- μια οντολογία καθορίζει τις κοινές λέξεις και έννοιες (ή σημασίες) που χρησιμοποιούνται για να περιγράψουν και να αντιπροσωπεύσουν έναν τομέα της γνώσης
- μια οντολογία είναι ένα προϊόν εφαρμοσμένης μηχανικής που αποτελείται από ένα συγκεκριμένο λεξιλόγιο που χρησιμοποιείται για να περιγράψει [ένα μέρος] την πραγματικότητα, μαζί με ένα σύνολο ρητών υποθέσεων σχετικά με την προοριζόμενη έννοια αυτού του λεξιλογίου – με άλλα λόγια, την προδιαγραφή μιας συνολικής σύλληψης για το υπό περιγραφή πεδίο.

Ο πρώτος καθορισμός έχει δύο μέρη:

- περιγραφή και αντιπροσώπευση ένός τομέα γνώσης
- καθορισμός των κοινών λέξεων και εννοιών της περιγραφής.

Επομένως, μια περιγραφή είναι ή μπορεί να είναι μια οντολογία. Γενικά μια οντολογία περιλαμβάνει τα εξής :

- κατηγορίες (classes) για τις εκάστοτε περιοχές των περιπτώσεων ενδιαφέροντος

- στιγμιότυπα (instances) δηλαδή ιδιαίτερες εκφάνσεις των κατηγοριών
- σχέσεις μεταξύ των κατηγοριών ή των στιγμιοτύπων τους
- ιδιότητες (και ιδιότητες τιμών) που μπορούν να λάβουν οι κατηγορίες και τα στιγμιότυπά τους
- λειτουργίες και διαδικασίες που σχετίζονται με τα παραπάνω
- περιορισμούς και κανόνες που σχετίζονται επίσης με όσα αναφέρθηκαν

Εκτός από την περιγραφή ενός τομέα της γνώσης, είναι εξίσου σημαντικό να δημιουργήσουμε μια αναπαράσταση αυτού. Η αναπαράσταση σημαίνει ότι κωδικοποιούμε την περιγραφή με έναν τρόπο που επιτρέπει σε κάποιο να χρησιμοποιήσει την περιγραφή. Μια περιγραφή αποτελείται από τις λέξεις και τις φράσεις σε μια φυσική γλώσσα, δηλαδή από ένα λεξιλόγιο/ορολογία και προτάσεις που συνδυάζουν τις ορολογίες για να εκφράσουν τις σχέσεις μεταξύ των όρων. Έτσι να αναπαραστήσουμε σημαίνει να δομήσουμε την περιγραφή χρησιμοποιώντας τους όρους και τις προτάσεις. Καθορίζοντας τους όρους και συνδυάζοντας τους με συγκεκριμένους τρόπους διαμορφώνουμε την έννοια που επιθυμούμε για τον συγκεκριμένο τομέα γνώσης.

Στην Πληροφορική εντούτοις, χρησιμοποιούμε μια ελαφρώς πιό περίπλοκη ιδέα για να διαμορφώσουμε μια ακριβή και χρήσιμη αναπαράσταση. Δημιουργούμε ένα πρότυπο που το λογισμικό θα είναι σε θέση να χρησιμοποιήσει. Αναπαριστούμε τις κατηγορίες, τα στιγμιότυπα, τις σχέσεις, τις ιδιότητες, και τους κανόνες για τον τομέα γνώσης. Χρησιμοποιούμε τους όρους της φυσικής γλώσσας περιγραφής σαν ετικέτες για τις υπονοούμενες έννοιες- δηλαδή για να σημάνουμε εννοιολογικά τις κατηγορίες, τις ιδιότητες, και τις σχέσεις. Ουσιαστικά, αναπαριστούμε ή κωδικοποιούμε την οντολογία σε μια γλώσσα λογικής προκειμένου να αποφύγουμε ενδεχόμενες διφορούμενες ερμηνείες που ελλοχεύουν στην εκφορά της φυσικής γλώσσας αλλά και για να ορίσουμε σαφώς μια ιεραρχία και μια δομή στο πεδίο αναπαράστασης. Επιδιώκουμε προφανώς με αυτό τον τρόπο να καταστήσουμε τις αναπαριστάμενες έννοιες διαθέσιμη για χρήση από μηχανές με αντίστοιχο λογισμικό

Ο δεύτερος ορισμός αφορά την σύλληψη για το υπό περιγραφή πεδίο. Όταν συλλαμβάνουμε τον κόσμου ή ενός μέρους του κόσμου, έχουμε κατά νου, κυριολεκτικά, ένα διανοητικό πρότυπο εκείνου του μέρους του κόσμου. Για το παράδειγμα, όταν συλλαμβάνουμε τις επισκευές αυτοκινήτων, έχουμε πιθανώς ένα σύνολο διανοητικών εικόνων των αυτοκινήτων, των υποσυστημάτων και των μερών τους, ενός γκαράζ αυτοκινήτων ή ενός καταστήματος επισκευής, μηχανικών με στολές, και τα λοιπά. Εάν επρόκειτο να περιγράψουμε αυτές τις εικόνες, θα λειτουργούσαμε σύμφωνα με τον πρώτο

ορισμό δηλαδή θα ορίζαμε τους όρους των πραγμάτων που είναι σημαντικά για την έννοια των επισκευών αυτοκινήτων, τις ιδιότητες και τις σχέσεις τους. Λαμβάνοντας υπόψη τις ιδιαιτερότητες του κάθε συγκεκριμένου μέρους του κόσμου (μιας θεματικής περιοχής), με άλλα λόγια μιας σύλληψης επιδιώκουμε να το περιγράψουμε σε μας ή ένα άλλο πρόσωπο με έναν αρκετά λεπτομερή και ακριβή τρόπο, δηλαδή να το διευκρινίσουμε.

TERM	DEFINITION
Common Logic (CL)	The name for the ISO standard knowledge representation language based on KIF.
Cyc and OpenCyc	The first ontology-based knowledge representation system, whose development began in 1984 at the Microelectronics & Computer Technology Corporation (MCC) in Austin, Texas, and that is now commercialized by Cycorp (http://www.cyc.com/). Cyc is a repository of machine-interpretable commonsense knowledge represented as mostly first-order predicate logic-based ontologies (with some second-order logic extensions) and a reasoning engine. OpenCyc is the open source version of the Cyc technology (http://www.opencyc.org/).
DAML+OIL	DARPA (Defense Advanced Research Program) Agent Markup Language-Ontology Inference Layer: These are two XML- and Web-based languages to support the Semantic Web, which have recently fused. DAML originated from a US DARPA-sponsored program; OIL originated from a European Union-sponsored program. Together they constitute the most semantically expressive language available for WWW documents. The combined language is now supported by the W3C Web standards consortium.
Description logic	A knowledge representation formalism (sometimes called a terminological logic, classification logic, concept logic, or term subsumption logic) based on a subset of first-order predicate logic that is a declarative formalism for the representation and expression of knowledge and sound, tractable reasoning methods founded on a firm theoretical (logical) basis.
Frame-based knowledge representation	A knowledge representation formalism for expressing ontological information derived originally from the artificial intelligence (AI) language called KL-1, which itself is one of the earliest formalizations of the notion of semantic network. The notion of a frame comes from the early LISP programming language terminology used by early KR languages. In frame terminology, a concept is a class, and a relation is a slot. Attributes (sometimes called properties) are just slots defined on a domain (a specific class subtree) or one of its subdomains (a subclass of a domain class).

TERM	DEFINITION
Knowledge Interchange Format (KIF)	This is a knowledge representation language developed prior to the emergence of the Semantic Web. KIF is based on first-order predicate logic, with a slight extension into second-order logic (because it includes quantifying over sequence variables, i.e., predicates in a sequence). KIF has a LISP-like syntax. There is now an ISO-KIF standard called Common Logic (CL). CL has multiple syntaxes, including an XML and a Conceptual Graph syntax.
Open Knowledge Base Connectivity (OKBC) language	This is a language for knowledge access and interchange (an API) derived from the Generic Frame Protocol, developed in the early 1990s by knowledge representation technologists under the support of the DARPA Knowledge Sharing Effort (Patil et al., 1992). This protocol became the OKBC under the support of the DARPA High Performance Knowledge Base (HPKB) program, 1996–1999 (Cohen et al., 1998).
Ontolingua	A knowledge representation language based on KIF that is used for expressing ontologies (http://www.ksl.stanford.edu/software/ontolingua/). See Gruber (1993). Ontolingua was the first ontology language. It is currently also an ontology system with reasoning methods and is supported by the Knowledge Systems Laboratory at Stanford University.
Ontology	An ontology models the vocabulary and meaning of domains of interest: the objects (things) in domains; the relationships among those things; the properties, functions, and processes involving those things; and constraints on and rules about those things.
Protégé	Protégé is an ontology management tool developed and maintained by the Medical Informatics Laboratory at Stanford University based on the OKBC knowledge model and is recognized as an exemplary tool for managing ontologies (http://protege.stanford.edu/).

TERM	DEFINITION
Resource Definition Framework/Schema (RDF/S)	These are two languages. The first (RDF) expresses instance-level semantic relations phrased in terms of a triple: <subject, verb, object>, i.e., <object1, relation1, object2>. The second (RDFS) expresses class-level relations describing acceptable instance-level relations.
OWL	Web Ontology Language (sometimes called Ontology Web Languagea language developed by the W3C's Web Ontology Working Group and intended to be the successor of DAML+OIL. OWL is the most expressive knowledge representation for the Semantic Web so far.

Πίνακας 2. 6

Ο πίνακας 2. 6 περιλαμβάνει κάποια σημαντικά στοιχεία ορισμών που αφορούν γενικά στην τεχνολογία των οντολογιών αλλά και στην λογισμική υποστήριξή τους από τις σύγχρονες τεχνολογίες.

2.3.2 Η σύνταξη, η δομή και η σημασιολογία

2.3.2.1 Σύνταξη

Η σύνταξη προσδιορίζεται συνήθως με τη μορφή, το σχήμα, και τη δομή, και έτσι θα την απεικονίσουμε εδώ. Κάθε γλώσσα έχει μια σύνταξη και μια σημασιολογία. Μια γλώσσα μπορεί να θεωρηθεί ένα επίσημο σύστημα που έχει ένα αλφάβητο ή ένα λεξιλόγιο καθορισμένο (ή και τα δύο), ένα σύνολο κανόνων για το πως το αλφάβητο και το λεξιλόγιο μπορούν να συνδυαστούν στις νόμιμες δηλώσεις ή τις προτάσεις στη γλώσσα και έπειτα μια σημασιολογία για το αλφάβητο, το λεξιλόγιο, και τις δηλώσεις/τις προτάσεις. Θα μιλήσουμε πρώτα για τη σύνταξη προκειμένου να καταλήξουμε στη συζήτηση σχετικά με τη σημασιολογία (και την πραγματολογία). Η γλώσσα προγραμματισμού, ακριβώς όπως μια φυσική γλώσσα, έχει μια επίσημη σύνταξη.

Έτσι η σύνταξη είναι για τη τάξη και τη δομή. Εάν ένα πρόγραμμα που αναπτύσσουμε είναι συντακτικά σωστό, τότε ο μεταγλωττιστής- που αναλύει αρχικά και επιβεβαιώνει κατόπιν την σύνταξη – δεν θα παράγει μηνύματα προειδοποίησης ή λάθους. Όταν όλα τα συντακτικά λάθη (λάθη στη μορφοποίηση, στις εντολές, και στην απλή

δακτυλογράφηση) έχουν διορθωθεί, τότε ο μεταγλωττιστής θα συνεχίσει το υπόλοιπο της εργασίας του, με την δημιουργία των πινάκων συμβόλων, τους μετασχηματισμούς, και τις βελτιστοποιήσεις του κώδικα δηλαδή θα αρχίσει να εξετάζει την στοιχειώδη σημασιολογία του προγράμματος. Παραμένει στοιχειώδης η εξέταση αυτή,καθώς ο μεταγλωττιστής δεν ξέρει ποιος είναι ο σκοπός και ποια η ιδιαίτερη λογική ροή του προγράμματος, το οποίο είναι και ο λόγος για τον οποίο αυτά τα είδη σημασιολογικών λαθών προγραμματισμού καλούνται μερικές φορές λογικά λάθη. Αυτήν την περίοδο, αυτό το επίπεδο ελέγχου απαιτεί έναν άνθρωπο ή έναν εξαιρετικά έξυπνο μεταγλωττιστή (ίσως καθοδηγούμενο από ένα έμπειρο σύστημα που χρησιμοποιεί τη γνώση) για να ανιχνεύσει και να διορθώσει τα σημασιολογικά λάθη.

Στον κόσμο του Διαδικτύου, εξετάζουμε κατά κανόνα τα έγγραφα ως προς την σύνταξή τους με βάση τις γλώσσες σήμανσης και τα στοιχεία δομών. Η σύνταξη των εγγράφων περιλαμβάνει τις σειρές των χαρακτήρων από κάποιο αλφάβητο (για το κείμενο) ή κάποιο σύνολο καθορισμένων δυαδικών κωδικοποιήσεων (για τη γραφική παράσταση, το βίντεο, και τα λοιπά). Τα συντακτικά σύμβολα είναι χωρίς νόημα εκτός αν τους δίνεται μια σημασιολογική ερμηνεία, με άλλα λόγια, μια αντιστοίχιση σε αντικείμενα με βάση το πρότυπο όπου εκείνη η έννοια αντιπροσωπεύεται. Η σημασιολογική ερμηνεία είναι σημασιολογία: Ερμηνεύει τα συντακτικά σύμβολα όσον αφορά την προοριζόμενη έννοιά τους.

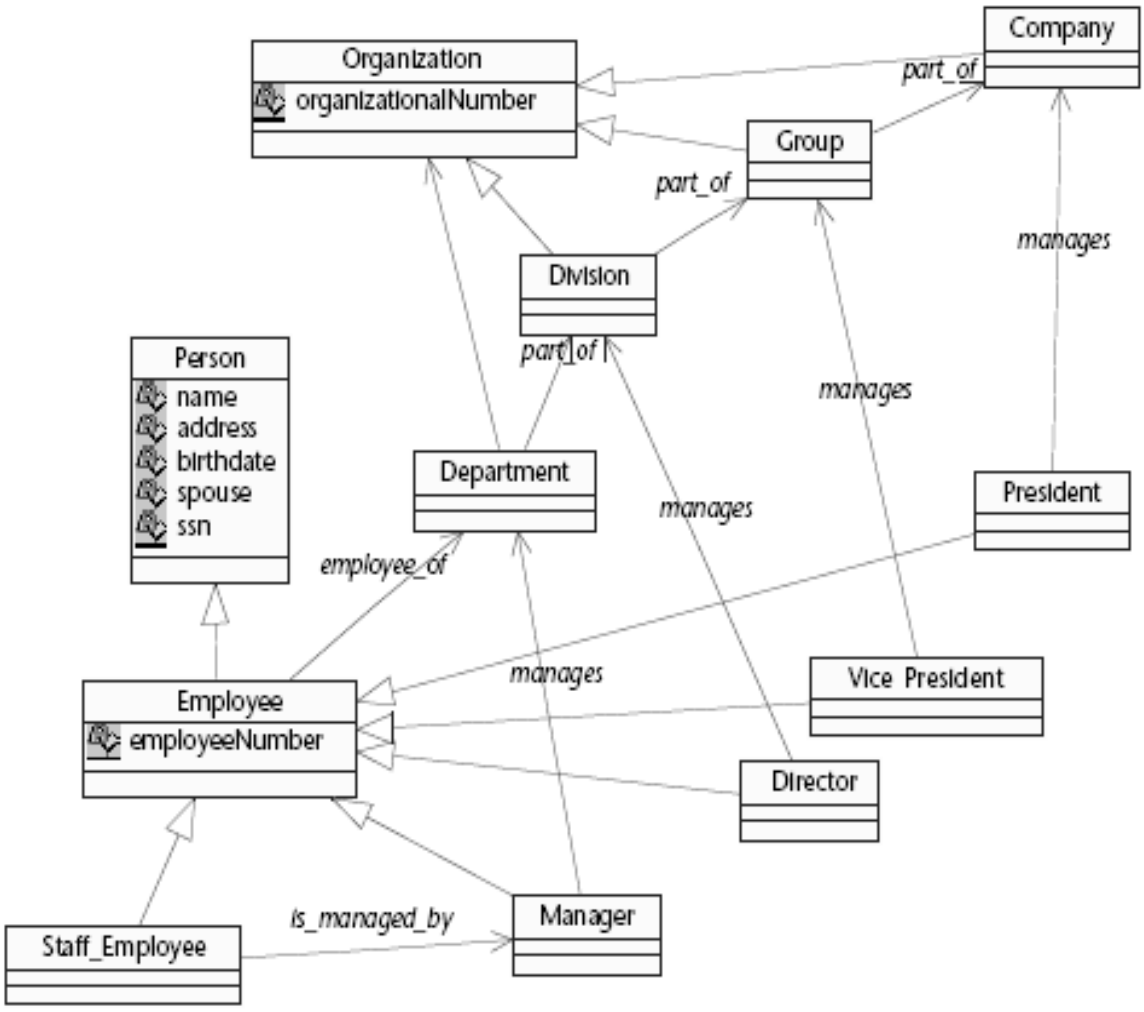
2.3.2.2 Δομή

Η σύνταξη είναι μορφή και σχήμα, αλλά είναι επίσης και δομή. Οι βάσεις δεδομένων, τα αντικείμενα του Διαδικτύου, τα αντικείμενα στον επερχόμενο σημασιολογικό Ιστό, τα μοντέλα και οι οντολογίες απαιτούν μια συγκεκριμένη δομή. Τα μοντέλα απαιτούν δομή, σαν ένα τρόπο για να χαρακτηρίζονται και να διαχωρίζονται τα στοιχεία του μοντέλου. Ένα σχήμα βάσεων δεδομένων, παραδείγματος χάριν, είναι πρώτιστα ένας τρόπος και να περιγραφεί και να οριστεί η δομή μιας βάσης δεδομένων. Με τον ορισμό, εννοούμε ότι τα αντικείμενα στη βάση δεδομένων - πινακες, στήλες, σειρές, και τιμές- απαιτείται να συμμορφώνονται με την δομή του σχήματος, ως προς τον τρόπο που τα στοιχεία του σχήματος οργανώνονται και άλλοτε περιλαμβάνονται μέσα σε άλλα στοιχεία. Αυτή η «διαταγή» επιβάλλεται από το σύστημα διαχείρισης βάσεων δεδομένων (DBMS (DataBase Management System)), το οποίο μπορούμε να σκεφτούμε, τουλάχιστον μερικώς, ως ένα διερμηνέα ή validator των στοιχείων που συνιστούν το

σχήμα βάσεων δεδομένων. Γενικά τα DBMSs είναι σε θέση να κάνουν πολύ περισσότερα από αυτό. Ωστόσο στην συγκεκριμένη περίπτωση το DBMS λειτουργεί σαν ένας parser/validator XML, που αναλύει τα αρχεία XML (έλεγχος σχετικά με την συντακτική προδιαγραφή της προς εξέταση έκδοσης XML) και τα έπειτα επικυρώνει με βάση μια ιδιαίτερη δομή-σύνταξη DTD (Document Type Definition) ή ένα σχήμα βασισμένο σε XML schema.

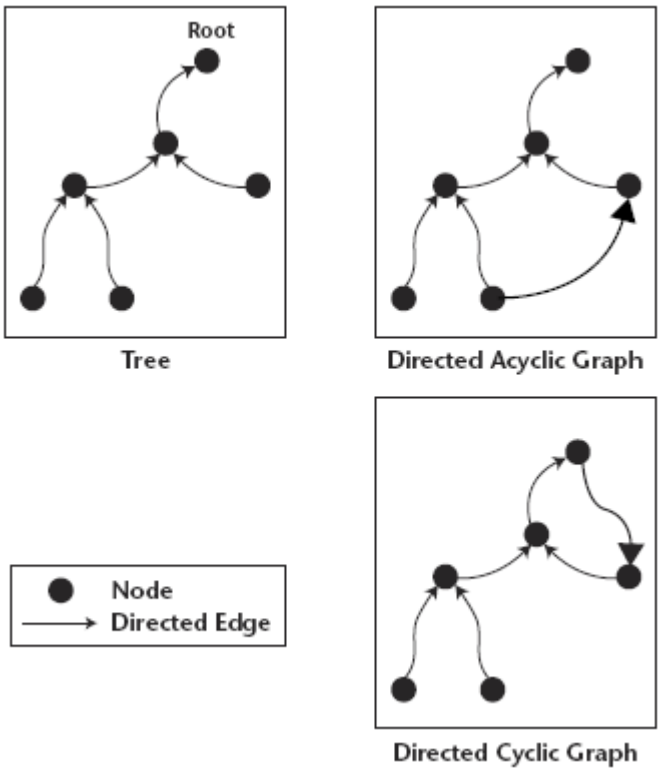
Τα εννοιολογικά πρότυπα, όπως αυτά που περιγράφονται σε UML, σχετίζονται επίσης με την δομή. Η δομή στα εννοιολογικά πρότυπα απεικονίζεται μερικώς στις ιεραρχίες κληρονομιάς μεταξύ των υποκατηγοριών: Μια κατηγορία είναι μια υποκατηγορία μιας άλλης κατηγορίας. Η δομή απεικονίζεται επίσης και στον ορισμό των σχέσεων : Μια κατηγορία είναι ένα μέρος ή αποτελεί μια άλλη κατηγορία. Η δομή απεικονίζεται και σε άλλες αυθαίρετες σχέσεις. Σε ένα πρότυπο UML μιας εφαρμογής ανθρώπινου δυναμικού παραδείγματος χάριν, δύο κατηγορίες μπορούν να είναι μέρη της σχέσης employee-of (παρόμοια με τη σχέση σε ένα εννοιολογικό σχήμα βάσεων δεδομένων, το οποίο κατασκευάζεται σε ένα πρότυπο οντοτήτων-συσχετίσεων), όπως φαίνεται χαρακτηριστικά στο σχήμα 2. 7. Σημειώστε ότι αυτό είναι κατά προσέγγιση το UML ισοδύναμο της οντολογίας OKBC στο σχήμα 2. 4 (εν τούτοις χωρίς την ελλοχεύουσεα λογική αφθονία και την ακρίβεια περιγραφής που παρατηρήσαμε στην οντολογία – που της επιτρέπει και την εξαιρετικά σημαντικά σημασιολογική ερμηνευσιμότητα από τις μηχανές).

Η δομή μπορεί να αντιπροσωπευθεί από μια γραφική σημειωση κόμβου και ακμής (node and edge), χρησιμοποιώντας δηλαδή έναν κόμβο και μια ακμή ή μια σύνδεση, η οποία μπορεί να είναι και κατευθυνόμενη. Η γενική μελέτη τέτοιων προτύπων κόμβου και ακμής καλείται θεωρία γράφων. Ένας γράφος είναι μια πιό περίπλοκη δομή δεδομένων από ένα δέντρο. Κι αυτό επειδή είναι ένα δέντρο που διαθέτει κατευθυνόμενες ή μη-κατευθυνόμενες συνδέσεις μεταξύ των κόμβων, δεν διαθέτει ρίζα και κανένα περιορισμό στο ότι από κάποιο κόμβο πρέπει να οδηγούμαστε οπωσδήποτε σε όλους τους άλλους κόμβους – όπως χαρακτηριστικά συμβαίνει με την ρίζα.



Σχήμα 2. 7 UML διάγραμμα ανθρωπινου δυναμικού εταιρείας

Η δομή λοιπόν αν και σημαντική δεν αποτελεί τον πιό καθοριστικό παράγοντα για την σημασιολογική ερμηνεία των προτύπων. Η δομή είναι μια *παρενέργεια* του βαθμού σημασιολογικής ερμηνείας που απαιτείται. Η γνώση (όπως κωδικοποιείται στις οντολογίες) είναι η σχετικά σύνθετη συμβολική διαμόρφωση (αναπαράσταση) κάποιας πτυχής της περιοχής ενδιαφέροντος.



Σχήμα 2. 8 Δέντρα και Γράφοι

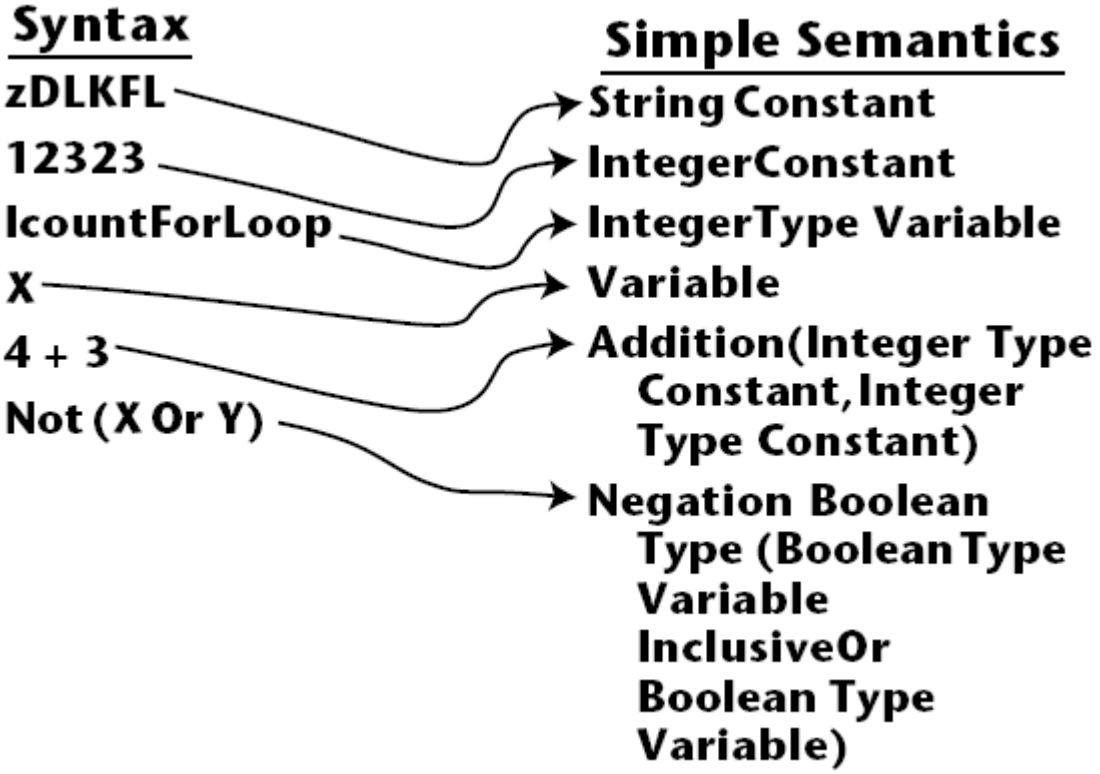
2.3.2.3 Σημασιολογία

Η σημασιολογική ερμηνεία είναι η χαρτογράφηση μεταξύ κάποιου δομημένου υποσυνόλου στοιχείων και ενός μοντέλου ενός συνόλου αντικειμένων σε μια περιοχή σχετικά με την προοριζόμενη έννοια εκείνων των αντικειμένων και τις μεταξύ τους σχέσεις.

Το μοντέλο βρίσκεται στο μυαλό του ανθρώπου. Ως άνθρωποι «καταλαβαίνουμε» τη σημασιολογία, που σημαίνει ότι αναπαριστούμε συμβολικά με κάποιο τρόπο τον κόσμο, τα αντικείμενα του κόσμου, και τις σχέσεις μεταξύ των αντικειμένων. Έχουμε τη σημασιολογία (κάποιο μέρος) του κόσμου στο μυαλό μας. Σε αυτή τη φάση ο κόσμος είναι πολύ δομημένος και ερμηνευμενός. Όταν βλέπουμε ένα έγγραφο κειμένου, βλέπουμε τα σύμβολα σε μια σελίδα και τα ερμηνεύουμε με βάση αυτό που σημαίνουν στο διανοητικό μας πρότυπό δηλαδή παρέχουμε τη σημασιολογία (σημασία). Εάν επιθυμούμε να βοηθήσουμε στη διάδοση της γνώσης που ενσωματώνεται σε ένα έγγραφο, καθιστάμε το έγγραφο διαθέσιμο σε άλλους ανθρώπους, αναμένοντας ότι θα χρησιμοποιήσουν το σημασιολογικό διερμηνέα τους (τα διανοητικά πρότυπά τους) και θα κατανοήσουν το νόημα από τα σύμβολα στις σελίδες εγγράφων. Έτσι, δεν υπάρχει καμία γνώση σε εκείνο το έγγραφο χωρίς κάποιο ή κάτι που να ερμηνεύει τη σημασιολογία του. Η

σημασιολογική ερμηνεία «δημιουργεί» τη γνώση από τα ειδάλλως χωρίς νόημα σύμβολα σε ένα κείμενο.

Εάν επιθυμούμε να κάνουμε τον υπολογιστή να βοηθήσει στη διάδοση της γνώσης που ενσωματώνεται σε ένα έγγραφο – που είναι και η πεμπτουσία του σημασιολογικού Ιστού – πρέπει να ικανοποιήσουμε την ανάγκη να αυτοματοποιηθεί η σημασιολογική διαδικασία ερμηνείας. Πρέπει να περιγράψουμε και να αναπαραστήσουμε με έναν υπολογιστικά χρησιμοποιήσιμο τρόπο ένα τμήμα των διανοητικών προτύπων μας για τις συγκεκριμένες περιοχές. Οι οντολογίες παρέχουν σε μας αυτή την ικανότητα. Το λογισμικό του μέλλοντος (συμπεριλαμβανομένων των ευφυών πρακτόρων, των υπηρεσιών Ιστού, και τα λοιπά) θα είναι σε θέση να χρησιμοποιήσει τη γνώση που κωδικοποιείται στις οντολογίες και τουλάχιστον μερικώς να καταλάβει και να ερμηνεύσει σημασιολογικά τα έγγραφα Ιστού και τα αντικείμενά μας.



Σχήμα 2. 9 Χαρτογράφηση μεταξύ σύνταξης και σημασιολογίας

2.3.3 Επίπεδα Αναπαράστασης

Κατά τη συζήτηση των οντολογιών, είναι σημαντικό να γίνουν κάποιες διακρίσεις μεταξύ των διάφορων επιπέδων αναπαράστασης. Αυτές οι διακρίσεις είναι απαραίτητες επειδή οι οντολογίες μπορούν να αντιμετωπισθούν ως γλώσσες, ή συντακτικά λεξιλόγια με τη συνοδεία σημασιολογίας. Επιπλέον, οι οντολογίες διαθέτουν γνωσιολογικό περιεχόμενο. Το περιεχόμενο τους μπορεί να εκφραστεί χρησιμοποιώντας μια γλώσσα περιεχομένου, η οποία καλείται συνήθως γλώσσα αναπαράστασης γνώσης. Επομένως μιλάμε για τουλάχιστον δύο επίπεδα αναπαράστασης :

- το γλωσσικό επίπεδο αναπαράστασης γνώσης, που καλείται και μετα - επίπεδο αναφορικά με το ελλοχεύον επίπεδο, και
- το επίπεδο αντικειμένου, το οποίο είναι το ελλοχεύον επίπεδο περιεχομένου, το επίπεδο στο οποίο οι οντολογίες εκφράζονται.

Αλλά οι έννοιες του μετα - επιπέδου και του επιπέδου αντικειμένου είναι σχετικές με τα ιδιαίτερα επίπεδα για τα οποία κάποιος μιλά. Γνωρίζουμε ότι χρειαζόμαστε ένα τρίτο επίπεδο, το επίπεδο στιγμιοτύπων. Αυτό είναι το επίπεδο στο οποίο τα στιγμιότυπα των κατηγοριών οντολογίας υπάρχουν.

Έτσι εάν αναφερόμαστε στο επίπεδο περίπτωσης, το επίπεδο αυτό μπορεί να αντιμετωπισθεί ως επίπεδο αντικειμένου, και το μετα – επίπεδό του είναι το επίπεδο στο οποίο η κατηγορία ή η καθολική γνώση βεβαιώνεται (οι έννοιες της οντολογίας). Εάν αντ' αυτού κάποιος ασχολείται με τις κατηγορίες ή το καθολικό επίπεδο, κατόπιν εκείνο το επίπεδο μπορεί να αντιμετωπισθεί ως επίπεδο αντικειμένου, και το μετα – επίπεδό του είναι το επίπεδο της γλώσσας αναπαράστασης γνώσης. Ο πίνακας 2. 10 επιδεικνύει τα τρία επίπεδα αναπαράστασης που απαιτούνται για τις οντολογίες και τα είδη κατασκευασμάτων που αντιπροσωπεύονται στο κάθε μεμονωμένο επίπεδο :

- Επίπεδο $1^{ο}$: Το επίπεδο αναπαράστασης γνώσης
- Επίπεδο $2^{ο}$: Το επίπεδο εννοιών της οντολογίας
- Επίπεδο $3^{ο}$:Το επίπεδο στιγμιοτύπων της οντολογίας

Το γλωσσικό επίπεδο αναπαράστασης γνώσης (το πιό υψηλό μετα - επίπεδο) καθορίζει τα κατασκευάσματα που θα χρησιμοποιηθούν στο επίπεδο έννοιας οντολογίας Αυτά τα κατασκευάσματα περιλαμβάνουν τις έννοιες της κατηγορίας, των σχέσεων, των ιδιοτήτων και τα λοιπά Τα παραδείγματα των γλωσσών KR περιλαμβάνουν τις γλώσσες που προηγήθηκαν του σημασιολογικού Ιστός. Τέτοιες γλώσσες είναι οι KL-ONE, Ontolingua, Classic, LOOM, Knowledge Interchange Format (KIF), CycL, Unified

Modeling Language (UML) - και οι σημασιολογικές γλώσσες Ιστού, συμπεριλαμβανομένων των RDF/S, DAML+OIL, και OWL.

LEVEL	EXAMPLE CONSTRUCTS
Knowledge representation (KR) language (Ontology Language) level: Meta level to the ontology concept level	Class, Relation, Instance, Function, Attribute, Property, Constraint, Axiom, Rule
Ontology concept (OC) level: Object level to the KR language level, meta level to the instance level	Person, Location, Event, Parent, Hammer, River, FinancialTransaction, BuyingAHouse, Automobile, TravelPlanning, etc.
Ontology instance (OI) level: Object level to the ontology concept level	Harry X. Landsford III, Ralph Waldo Emerson, Person560234, PurchaseOrder TransactionEvent6117090, 1995-96 V-6 Ford Taurus 244/4.0 Aerostar Automatic with Block Casting # 95TM-AB and Head Casting 95TM

Πίνακας 2. 10

Στο δεύτερο επίπεδο, το επίπεδο έννοιας οντολογίας (OC), οι οντολογίες καθορίζονται χρησιμοποιώντας τα κατασκευάσματα του επιπέδου KR. Σε αυτό το επίπεδο, ενδιαφερόμαστε για τη διαμόρφωση του γενικού ή καθολικού περιεχομένου.

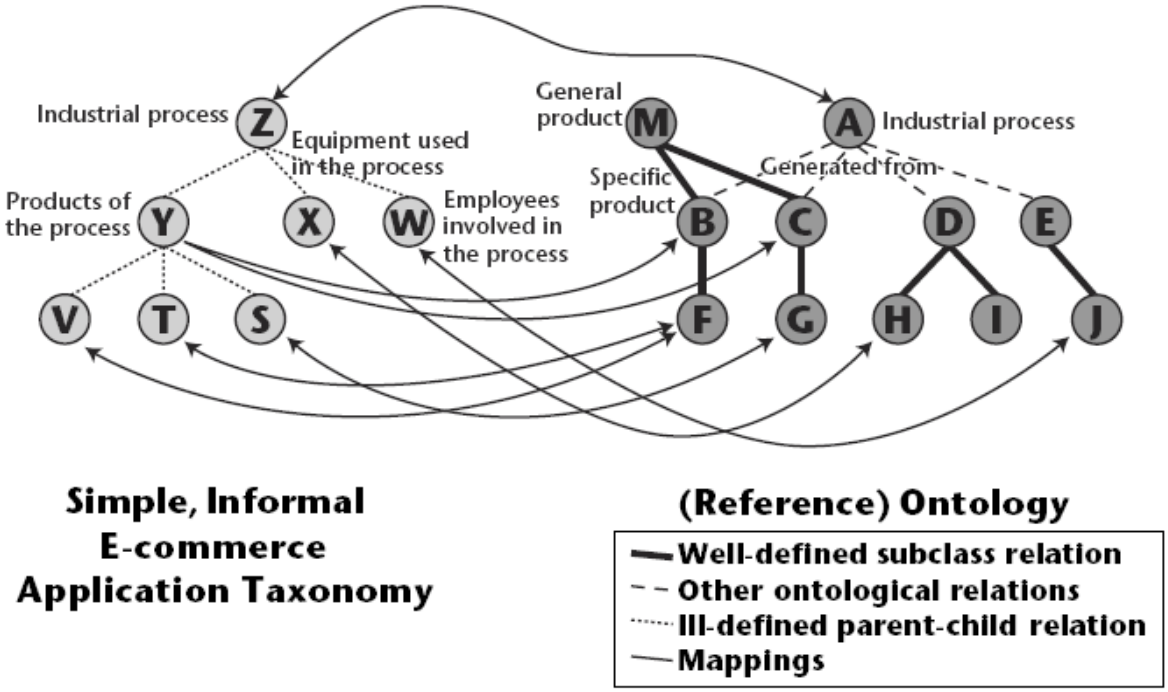
Στο τρίτο και χαμηλότερο επίπεδο, το επίπεδο περίπτωσης οντολογίας, τα κατασκευάσματα είναι στιγμιότυπα κατασκευασμάτων επιπέδων έννοιών οντολογίας. Έτσι αυτό το επίπεδο αφορά τη βάση γνώσεων ή τη βάση γεγονότος, τους ισχυρισμούς για τα στιγμιότυπα ή τα άτομα

2.3.4 Η οντολογία και το σημασιολογικό πρόβλημα χαρτογράφησης (mapping problem)

Ένα σημαντικό ζήτημα στην κατανόηση και την ανάπτυξη των οντολογιών είναι το οντολογικό ή σημασιολογικό πρόβλημα χαρτογράφησης. Λέμε «ή σημασιολογικό πρόβλημα χαρτογράφησης» επειδή αυτό είναι ένα ζήτημα που έχει επιπτώσεις σε οτιδήποτε στην Πληροφορική καλείται να αντιμετωπίσει σημασιολογικά προβλήματα – όπως είναι το πρόβλημα της αναπαράστασης της σημασίας για τα συστήματα, τις εφαρμογές, τις βάσεις δεδομένων, και τις συλλογές εγγράφων.

Αυτό το σημασιολογικό πρόβλημα υπάρχει με και χωρίς τις οντολογίες. Αυτό σημαίνει ότι υπάρχει μέσα σε οποιαδήποτε δεδομένη σημασιολογική αναπαράσταση όπως μια οντολογία. Μέσα σε μια οντολογία, θα πρέπει να εστιάσει κανείς σε ένα συγκεκριμένο πλαίσιο της οντολογίας, λαμβάνοντας υπόψη έναν συγκεκριμένο σκοπό ή μια λογική ή τη χρήση της οντολογίας. Και χωρίς τις οντολογίες, θα πρέπει να εστιάσουμε στη σημασιολογική ισοδυναμία μεταξύ των διαφορετικών εννοιών και των σχέσεων σε δύο ή περισσότερες διαφορετικές οντολογίες. Αυτές οι οντολογίες μπορεί να αναφέρονται σε περίπου ίδια πραγματα αλλά ενδεχόμενα τούτο να μην συμβαίνει. Το πιό πιθανό είναι ότι οι δύο ή περισσότερες διαφορετικές οντολογίες που πρέπει να χαρτογραφήσουμε θα περιγράφουν μια συγκεκριμένη περιοχή ενδιαφέροντος με τον ίδιο ή με κάποιο παρεμφερή τρόπο. Αυτό που επιβάλεται είναι να αποφύγουμε την σημασιολογική απώλεια πληροφορίας.

Στο σχήμα 2. 11 μπορούμε να δούμε μια χαρτογράφηση από μια οντολογία σε μια ταξονομία ηλεκτρονικού εμπορίου (παραδείγματος χάριν, μια μερίδα του προϊόντος UNSPSC). Στο δεξιό τμήμα του σχήματος βρίσκεται η οντολογία αναφοράς με τις σημασιολογικά καθορισμένες με σαφήνεια σχέσεις και στο αριστερό είναι η ταξονομία που χρησιμοποιείται για μια εφαρμογή ηλεκτρονικού εμπορίου με τις λιγότερο σαφώς καθορισμένες σχέσεις. Στην πράξη, μπορεί να πρέπει να διατηρήσουμε τις χαρτογραφήσεις μόνο μεταξύ οντολογιών (ή, όπως σε αυτό το παράδειγμα, μεταξύ των οντολογιών και των ταξονομιών) επειδή κάθε σύστημα αναπαράστασης γνώσης μπορεί να ρυθμιστεί από χωριστές διαδικασίες και πρέπει να εξελιχθεί ξεχωριστά. Γενικά, ο καθορισμός της σημασιολογικής ισοδυναμίας (χαρτογράφησης) μεταξύ των εννοιών σε δύο οντολογίες είναι σκληρός και απαιτεί την ανθρώπινη γνώση της σημασιολογίας των δύο πλευρών και την λήψη απόφασης (αν και τα τρέχοντα διοικητικά εργαλεία οντολογίας έχουν κάποια αυτοματοποιημένη υποστήριξη) για να γίνουν σωστές χαρτογραφήσεις. Αν και τα ονόματα (ετικέτες) δύο εννοιών μπορούν να είναι τα ίδια (ή απολύτως διαφορετικά) στις δύο οντολογίες, δεν υπάρχει καμία εγγύηση ότι εκείνες οι έννοιες σημαίνουν το ίδιο πράγμα (ή σημαίνουν διαφορετικά πράγματα). Έχουμε δει νωρίτερα ότι οι όροι (λέξεις ή ετικέτες) έχουν την πολύ αδύνατη σημασιολογία καθαυτοί. Η ταυτοποίηση γραμματοσειρών δεν μπορεί να αποτελεί εχέγγυο για σημασιολογική ταύτιση ή ισοδυναμία. Ομοίως, η δομική αντιστοιχία δεν μπορεί να στηρίξει την εξασφάλιση της σημασιολογικής αντιστοιχίας. Ο σημασιολογικός καθορισμός ισοδυναμιών και η δημιουργία των χαρτογραφήσεων μεταξύ οντολογιών θα παραμείνουν μια ημιαυτόματη διαδικασία για αρκετά χρόνια ακόμα.



Σχήμα 2. 11

Οι περισσότερες γλώσσες οντολογίας και τα συνοδευτικά εργαλεία τους έχουν κάποιες δυνατότητες για τον καθορισμό των χαρτογραφήσεων μεταξύ των οντολογιών. Ο απλούστερος μηχανισμός είναι ένα include ή import statement με το οποίο μια οντολογία περιλαμβάνει ή εισάγει μια άλλη οντολογία. Αυτός είναι ο απλούστερος μηχανισμός, επειδή φέρνει ολόκληρη την οντολογία αναφοράς στην τρέχουσα οντολογία και όλες οι έννοιες και οι σχέσεις της εισαγόμενης οντολογίας είναι διαθέσιμες στο νέο επεκταθέν σχήμα οντολογίας. Εντούτοις, αυτές οι νέες εισηγμένες έννοιες και σχέσεις δεν αποκαθίστανται σημασιολογικά (δηλαδή δεν έχουν την επιθυμητή σημασιολογία όσον αφορά στην προϋπάρχουσα οντολογία που τις περιλάμβανε. Μετά από την εισαγωγή, η οντολογία μπορεί να υποβληθεί σε μια σφιχτότερη σημασιολογική διόρθωση, με τη *συγχώνευση* ή τη *χαρτογράφηση* των παλαιών και νέων εννοιών και των σχέσεων. Η συγχώνευση θα οδηγήσει σε συγχωνευμένες έννοιες και σχέσεις. Η χαρτογράφηση θα κρατήσει τις έννοιες και τις σχέσεις και από τις δύο οντολογίες διακριτές αλλά συνδεδεμένες μεταξύ τους. Τα εργαλεία όπως Protege και Ontolingua/Chimaera υποστηρίζουν καλύτερα την διαδικασία συγχώνευσης από τη διαδικασία χαρτογράφησης. Οι σημασιολογικές γλώσσες Ιστού όπως DAML+OIL και η OWL έχουν τα κατασκευάσματα που επιτρέπουν την εισαγωγή σε μια οντολογία, αλλά και την δυνατότητα να δηλώσουν ότι δύο έννοιες (που αντιπροσωπεύονται ως κατηγορίες) είναι σημασιολογικά ταυτόσημες. Επιπλέον, η OWL επιτρέπει να δηλώσουμε ότι δύο άτομα

(στιγμιότυπα) είναι τα ίδια σημασιολογικά – ένα καλό βήμα προς τη μεγαλύτερη υποστήριξη για τη σημασιολογική χαρτογράφηση.

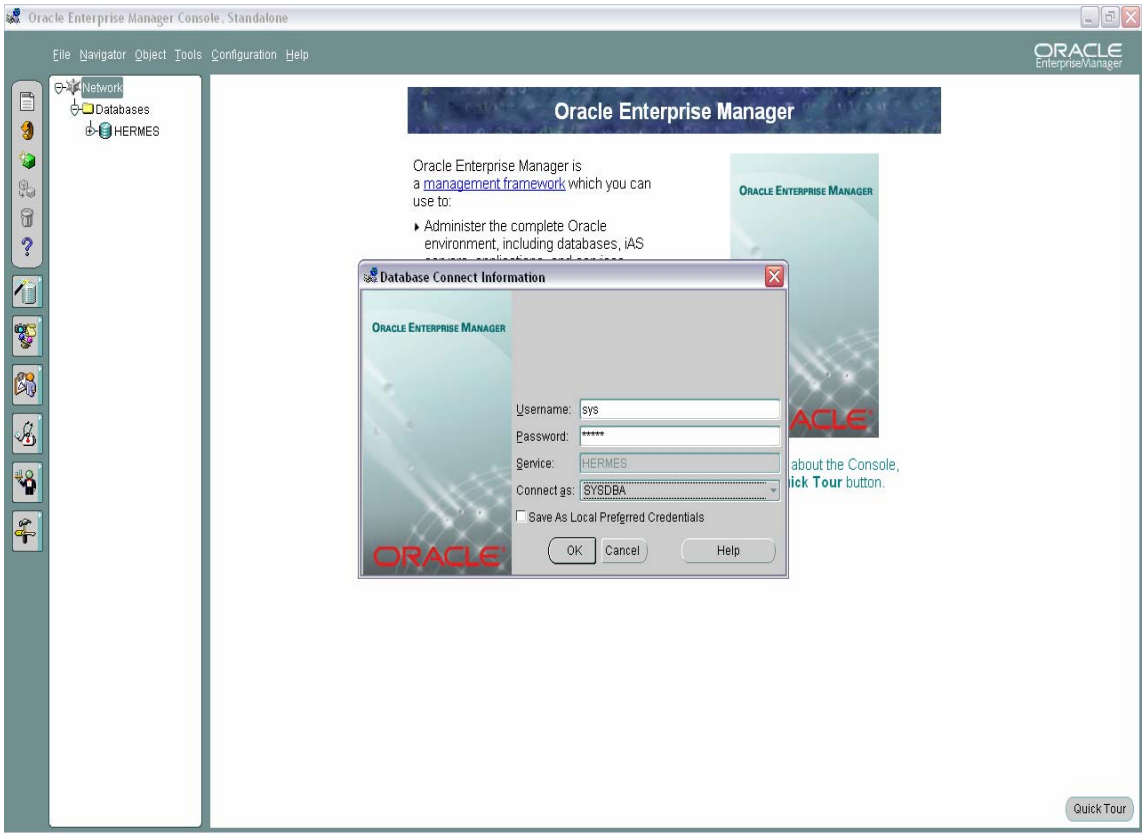
ΚΕΦΑΛΑΙΟ 3^{ο} : ΕΦΑΡΜΟΓΗ

3.1 Από το Oracle schema στον σημασιολογικό κόσμο

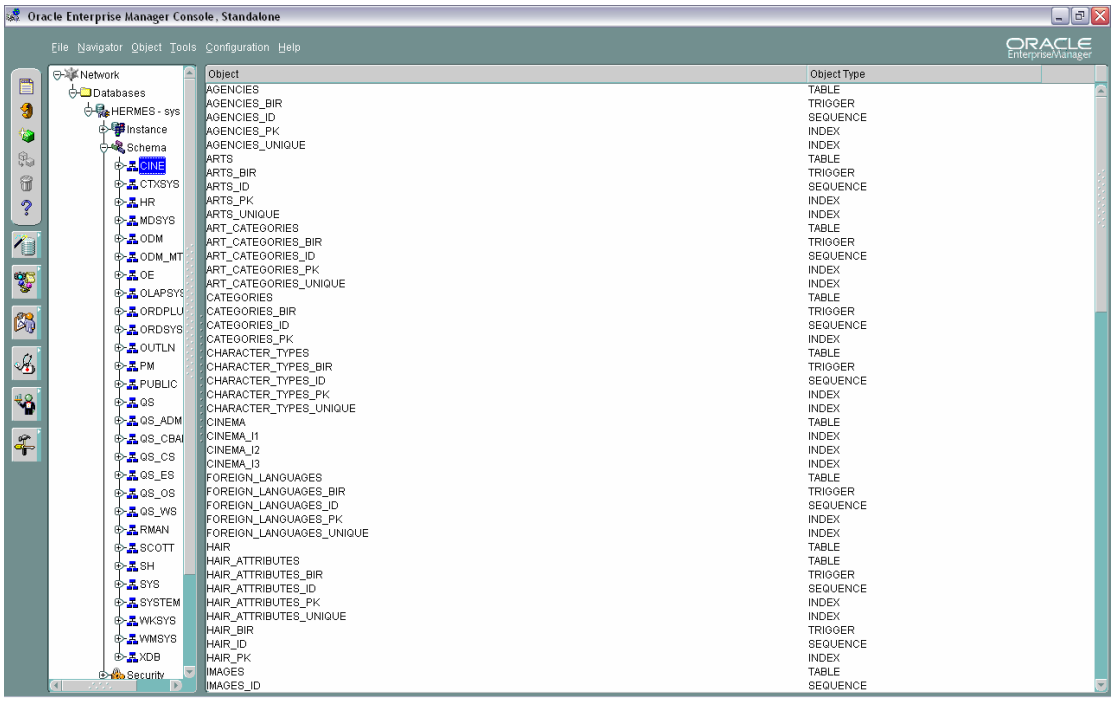
Το αρχικό μέλημα στην όλη προσπάθεια, ήταν η ανάκτηση γνώσης από μια βάση δεδομένων και η μετατροπή αυτής σε μια οντολογία. Η βάση δεδομένων που χρησιμοποιήθηκε ήταν η Oracle 9i και το schema της βάσης φαίνεται παρακάτω.

Αναφορικά με το schema πρέπει να τονιστεί ότι το θεματικό του περιεχόμενο αφορά μια βάση φωτογραφικού υλικού και video για κάποια κινηματογραφικά δρώμενα, πρακτορεία και μοντέλα. Η επιδίωξη ήταν αρχικά να μπορέσουμε να δούμε όλη την περιεχόμενη πληροφορία στη βάση σαν μια ενότητα και να επιτύχουμε ένα migration (όρος που είναι αδόκιμος για το εν λόγω πόνημα, αλλά κυρίως χρησιμοποιείται για μετακίνηση πληροφορίας μεταξύ βάσεων δεδομένων) ώστε να μην χαθεί η παραμικρή λεπτομέρεια. Κι αυτό διότι στο στάδιο αυτό δεν μπορούμε να κάνουμε μια a priori thin grain analysis αλλά θέλουμε να βρούμε μια τακτική για να πάμε στο πεδίο των οντολογιών χωρίς απώλειες. Ακολουθούν στις εικόνες 3. 1, 3. 2 και 3. 3 κάποια screenshots της βάσης όπως αυτή δόθηκε αρχικά. Κατόπιν παρατίθεται το oracle schema της βάσης με όλη την αρχική πληροφορία.

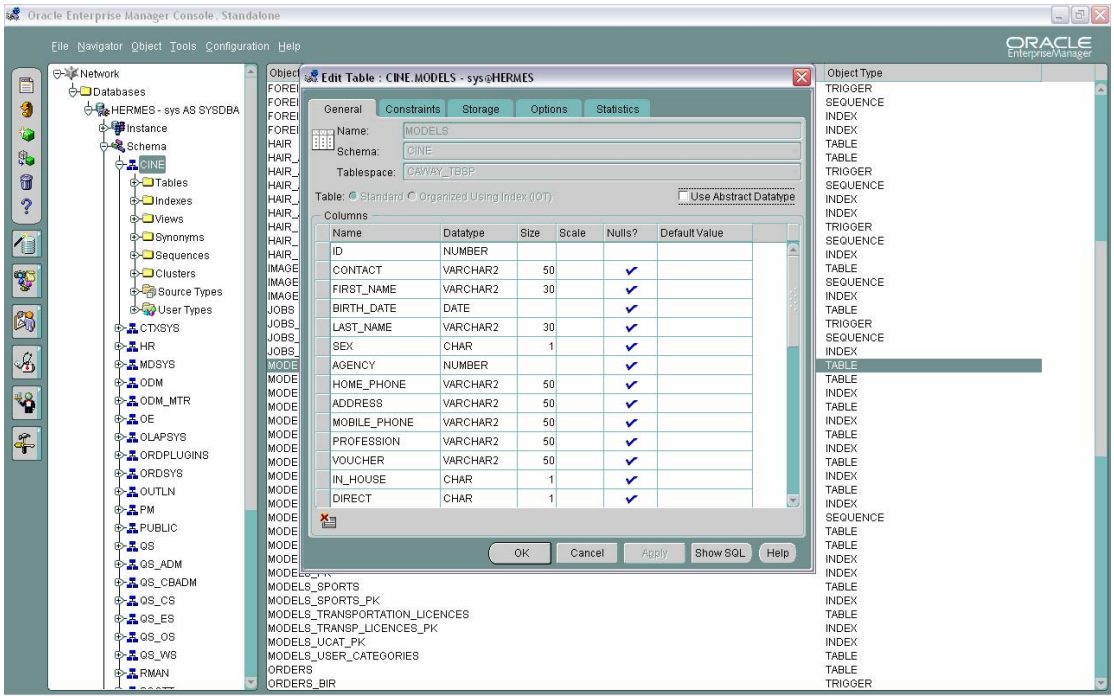
Εικόνα 3. 1



Εικόνα 3. 2



Εικόνα 3. 3



ORACLE SCHEMA

```
-----------------------------------------------------------------------
--                    CINEGRAM TABLESPACE                            --

-----------------------------------------------------------------------

--assuming you have an oracle db instance called CAWAY,you have oracle server installed on your C drive and your sys password is 'oracle' otherwise change accordingly--


connect sys/panos@HERMES as SYSDBA;

CREATE TABLESPACE CAWAY_TBSP
   LOGGING
   DATAFILE 'C:\ORANT_9. 2\ORADATA\HERMES\CAWAY_TBSP. ora' SIZE
   100M EXTENT MANAGEMENT LOCAL SEGMENT SPACE MANAGEMENT  AUTO;

-----------------------------------------------------------------------
--                    CINEGRAM DB USER                               --
                        --
-----------------------------------------------------------------------

CREATE USER CINE PROFILE DEFAULT IDENTIFIED BY cinegram
DEFAULT TABLESPACE CAWAY_TBSP
ACCOUNT UNLOCK;
```

```
GRANT CONNECT TO CINE;
GRANT CREATE TRIGGER TO CINE;
GRANT CREATE ANY DIRECTORY TO CINE;
GRANT SELECT ANY DICTIONARY TO CINE;
GRANT DROP ANY DIRECTORY TO CINE;
GRANT EXP_FULL_DATABASE TO CINE;
ALTER USER CINE DEFAULT ROLE ALL;
ALTER USER CINE
QUOTA UNLIMITED ON CAWAY_TBSP;

commit;

disconnect;

connect cine/cinegram@HERMES;
-------------------------------------------------------------
--                  USERS TABLE                  --
-------------------------------------------------------------
CREATE TABLE USERS (
ID NUMBER NOT NULL,
LOGIN VARCHAR2(10 byte) NOT NULL,
PASSWORD VARCHAR2(10 byte) NOT NULL,
ULEVEL NUMBER(2) NOT NULL,
CONSTRAINT USERS_PK PRIMARY KEY(ID)
);

-------------------------------------------------------------

-------------------------------------------------------------
--                  USER CATEGORIES TABLE                  --
-------------------------------------------------------------
CREATE TABLE USER_CATEGORIES (
ID NUMBER NOT NULL,
USER_ID NUMBER NOT NULL,
CATEGORY VARCHAR2(20),
CONSTRAINT USER_CAT_PK PRIMARY KEY(ID),
CONSTRAINT USER_FK FOREIGN KEY(USER_ID)
REFERENCES USERS(ID) ON DELETE CASCADE
);

-------------------------------------------------------------



-------------------------------------------------------------
--                  AGENCIES TABLE                  --
-------------------------------------------------------------
CREATE TABLE AGENCIES
(
 id         INTEGER,
 agency_name VARCHAR2 ( 20 ) NOT NULL,
 CONSTRAINT AGENCIES_PK PRIMARY KEY ( id ),
```

```
 CONSTRAINT AGENCIES_UNIQUE UNIQUE ( agency_name )
);
--------------------------------------------------------------------



--------------------------------------------------------------------
--                    TEETH TABLE                    --
--------------------------------------------------------------------
CREATE TABLE TEETH
(
 id         INTEGER,
 teeth_type VARCHAR2 ( 20 ) NOT NULL,
 CONSTRAINT TEETH_PK PRIMARY KEY ( id ),
 CONSTRAINT TEETH_UNIQUE UNIQUE ( teeth_type )
);
--------------------------------------------------------------------



--------------------------------------------------------------------
--                    MODELS TABLE                    --
--------------------------------------------------------------------


CREATE TABLE MODELS (ID NUMBER NOT NULL,
   CONTACT VARCHAR2(50 byte), FIRST_NAME VARCHAR2(30 byte),
   BIRTH_DATE DATE, LAST_NAME VARCHAR2(30 byte), SEX
   CHAR(1 byte), AGENCY NUMBER, HOME_PHONE VARCHAR2(50 byte),
   ADDRESS VARCHAR2(50 byte), MOBILE_PHONE VARCHAR2(50 byte),
   PROFESSION VARCHAR2(50 byte), VOUCHER VARCHAR2(50 byte),
   IN_HOUSE CHAR(1 byte), DIRECT CHAR(1 byte),HEIGHT
   VARCHAR2(8 byte), JEEN_NO VARCHAR2(8 byte), SHOE_NO
   VARCHAR2(8 byte), COSTUME_NO VARCHAR2(8 byte), DRESS_NO
   VARCHAR2(8 byte), BRA_NO VARCHAR2(8 byte), PANTS_NO
   VARCHAR2(8 byte), BODY_TYPE VARCHAR2(20 byte), FITNESS
   VARCHAR2(20 byte), PIERCING VARCHAR2(50 byte), TATTOO
   VARCHAR2(50 byte), EYES VARCHAR2(20 byte), EYE_COLOR
   VARCHAR2(20 byte), HANDS VARCHAR2(20 byte), DESCRIPTION
   VARCHAR2(400 byte), COUNTRY VARCHAR2(20 byte), TEETH
   NUMBER(10),  TATTOOCOMMENTS VARCHAR2(200),INPUT_DATE DATE,
   CONSTRAINT MODELS_FK FOREIGN KEY(AGENCY)
   REFERENCES AGENCIES(ID),
   CONSTRAINT MODELS_FK2 FOREIGN KEY(TEETH)
   REFERENCES TEETH(ID),
   CONSTRAINT MODELS_PK PRIMARY KEY(ID)
);

--------------------------------------------------------------------

--------------------------------------------------------------------
```

```
--                  MODELS-USER CATEGORIES TABLE                    --
-----------------------------------------------------------------------
CREATE TABLE MODELS_USER_CATEGORIES (
MODELS_ID NUMBER NOT NULL,
USER_CATEGORIES_ID NUMBER NOT NULL,
CONSTRAINT MODELS_UCAT_PK PRIMARY
KEY(USER_CATEGORIES_ID,MODELS_ID),
CONSTRAINT MODELS_FK3 FOREIGN KEY(MODELS_ID)
REFERENCES MODELS(ID) ON DELETE CASCADE,
CONSTRAINT UCAT_FK  FOREIGN KEY(USER_CATEGORIES_ID)
REFERENCES USER_CATEGORIES(ID) ON DELETE CASCADE
);

-----------------------------------------------------------------------


-----------------------------------------------------------------------
--                  JOBS TABLE                    --
-----------------------------------------------------------------------
CREATE TABLE JOBS
(
 id          INTEGER,
 job_number VARCHAR2( 10),
 job_name VARCHAR2 ( 100 ) NOT NULL,
 job_date DATE,
 CONSTRAINT JOBS_PK PRIMARY KEY ( id )
);
-----------------------------------------------------------------------


-----------------------------------------------------------------------


-----------------------------------------------------------------------
--                  TAPES TYPES TABLE                    --
-----------------------------------------------------------------------
CREATE TABLE TAPE_TYPES
(
 id          INTEGER,
 casette_type VARCHAR2 ( 10 ) NOT NULL,
 CONSTRAINT TAPE_TYPES_PK PRIMARY KEY ( ID ),
 CONSTRAINT TAPE_TYPES_UNIQUE UNIQUE ( casette_type )
);
-----------------------------------------------------------------------


-----------------------------------------------------------------------
--              FOREIGN_LANGUAGES TABLE                    --
-----------------------------------------------------------------------
CREATE TABLE FOREIGN_LANGUAGES
(
 id      INTEGER,
 language VARCHAR2 ( 20 ) NOT NULL,
```

```
 CONSTRAINT FOREIGN_LANGUAGES_PK PRIMARY KEY ( id ),
 CONSTRAINT FOREIGN_LANGUAGES_UNIQUE UNIQUE ( language )
);
-----------------------------------------------------------------


-----------------------------------------------------------------
--              MODELS_FOREIGN_LANGUAGES TABLE              --
-----------------------------------------------------------------
CREATE TABLE MODELS_FOREIGN_LANGUAGES
(
 models_id          INTEGER,
 foreign_languages_id INTEGER,
 CONSTRAINT MODELS_FOREIGN_LANGUAGES_PK PRIMARY KEY ( models_id,
foreign_languages_id ),
 CONSTRAINT MODELS_FOREIGN_LANGUAGES_FK1 FOREIGN KEY ( models_id )
REFERENCES MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_FOREIGN_LANGUAGES_FK2 FOREIGN KEY (
foreign_languages_id ) REFERENCES FOREIGN_LANGUAGES ( id ) ON DELETE
CASCADE
);
-----------------------------------------------------------------


-----------------------------------------------------------------
--                    SPORTS TABLE                         --
-----------------------------------------------------------------
CREATE TABLE SPORTS
(
 id    INTEGER,
 sport VARCHAR2 ( 20 ) NOT NULL,
 CONSTRAINT SPORTS_PK PRIMARY KEY ( id ),
 CONSTRAINT SPORTS_UNIQUE UNIQUE ( sport )
);
-----------------------------------------------------------------


-----------------------------------------------------------------
--              MODELS_SPORTS TABLE                        --
-----------------------------------------------------------------
CREATE TABLE MODELS_SPORTS
(
 models_id INTEGER,
 sports_id INTEGER,
 CONSTRAINT MODELS_SPORTS_PK PRIMARY KEY ( models_id, sports_id ),
 CONSTRAINT MODELS_SPORTS_FK1 FOREIGN KEY ( models_id ) REFERENCES
MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_SPORTS_FK2 FOREIGN KEY ( sports_id ) REFERENCES
SPORTS ( id ) ON DELETE CASCADE
);
-----------------------------------------------------------------


-----------------------------------------------------------------
--              TRANSPORTATION_LICENCE TABLE               --
```

```
-------------------------------------------------------------------
CREATE TABLE TRANSPORTATION_LICENCES
(
 id               INTEGER,
 transportation_mean VARCHAR2 ( 30 ) NOT NULL,
 CONSTRAINT TRANSPORTATION_LICENCES_PK PRIMARY KEY ( id ),
 CONSTRAINT TRANSPORTATION_LICENCES_UNIQUE UNIQUE ( transportation_mean
)
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--               MODELS_TRANSPORTATION_LICENCE TABLE                --
-------------------------------------------------------------------
CREATE TABLE MODELS_TRANSPORTATION_LICENCES
(
 models_id                INTEGER,
 transportation_licences_id INTEGER,
 CONSTRAINT MODELS_TRANSP_LICENCES_PK PRIMARY KEY ( models_id,
transportation_licences_id ),
 CONSTRAINT MODELS_TRANSP_LICENCES_FK1 FOREIGN KEY ( models_id )
REFERENCES MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_TRANSP_LICENCES_FK2 FOREIGN KEY (
transportation_licences_id )  REFERENCES TRANSPORTATION_LICENCES ( id ) ON
DELETE CASCADE
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--               HAIR_ATTRIBUTES TABLE                      --
-------------------------------------------------------------------
CREATE TABLE HAIR_ATTRIBUTES
(
 id           INTEGER,
 hair_attribute VARCHAR2 ( 10 ) NOT NULL,
 CONSTRAINT HAIR_ATTRIBUTES_PK PRIMARY KEY ( id ),
 CONSTRAINT HAIR_ATTRIBUTES_UNIQUE UNIQUE ( hair_attribute )
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--                    HAIR TABLE                        --
-------------------------------------------------------------------
CREATE TABLE HAIR
(
 id              INTEGER,
 hair            VARCHAR2 ( 20 ) NOT NULL,
 hair_attributes_id INTEGER        NOT NULL,
 CONSTRAINT HAIR_PK PRIMARY KEY ( id ),
 CONSTRAINT HAIR_FK FOREIGN KEY ( hair_attributes_id ) REFERENCES
HAIR_ATTRIBUTES ( id ) ON DELETE CASCADE
```

```
);
-------------------------------------------------------------------


-------------------------------------------------------------------
--                    MODELS_HAIR TABLE                          --
-------------------------------------------------------------------
CREATE TABLE MODELS_HAIR
(
 models_id INTEGER,
 hair_id  INTEGER,
 CONSTRAINT MODELS_HAIR_PK PRIMARY KEY ( models_id, hair_id ),
 CONSTRAINT MODELS_HAIR_FK1 FOREIGN KEY ( models_id ) REFERENCES
MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_HAIR_FK2 FOREIGN KEY ( hair_id ) REFERENCES HAIR ( id )
ON DELETE CASCADE
);
-------------------------------------------------------------------


-------------------------------------------------------------------
--                CHARACTER_TYPES TABLE                          --
-------------------------------------------------------------------
CREATE TABLE CHARACTER_TYPES
(
 id           INTEGER,
 character_type VARCHAR2 ( 20 ) NOT NULL,
 CONSTRAINT CHARACTER_TYPES_PK PRIMARY KEY ( id ),
 CONSTRAINT CHARACTER_TYPES_UNIQUE UNIQUE ( character_type )
);
-------------------------------------------------------------------


-------------------------------------------------------------------
--            MODELS_CHARACTER_TYPES TABLE                       --
-------------------------------------------------------------------
CREATE TABLE MODELS_CHARACTER_TYPES
(
 models_id         INTEGER,
 character_types_id INTEGER,
 CONSTRAINT MODELS_CHARACTER_TYPES_PK PRIMARY KEY ( models_id,
character_types_id ),
 CONSTRAINT MODELS_CHARACTER_TYPES_FK1 FOREIGN KEY ( models_id )
REFERENCES MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_CHARACTER_TYPES_FK2 FOREIGN KEY ( character_types_id
) REFERENCES CHARACTER_TYPES ( id ) ON DELETE CASCADE
);
-------------------------------------------------------------------


-------------------------------------------------------------------
--                ARTS_CATEGORIES TABLE                          --
-------------------------------------------------------------------
CREATE TABLE ART_CATEGORIES
(
```

```
 id          INTEGER,
 art_category VARCHAR2 ( 20 ) NOT NULL,
 CONSTRAINT ART_CATEGORIES_PK PRIMARY KEY ( id ),
 CONSTRAINT ART_CATEGORIES_UNIQUE UNIQUE ( art_category )
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--                    ARTS TABLE                          --
-------------------------------------------------------------------
CREATE TABLE ARTS
(
 id            INTEGER,
 art           VARCHAR2 ( 20 ) NOT NULL,
 arts_category_id INTEGER        NOT NULL,
 CONSTRAINT ARTS_PK PRIMARY KEY ( id ),
 CONSTRAINT ARTS_FK FOREIGN KEY ( arts_category_id ) REFERENCES
ART_CATEGORIES ( id ) ON DELETE CASCADE,
 CONSTRAINT ARTS_UNIQUE UNIQUE ( art )
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--                  MODELS_ARTS TABLE                     --
-------------------------------------------------------------------
CREATE TABLE MODELS_ARTS
(
 models_id INTEGER,
 arts_id   INTEGER,
 CONSTRAINT MODELS_ARTS_PK PRIMARY KEY ( models_id, arts_id ),
 CONSTRAINT MODELS_ARTS_FK1 FOREIGN KEY ( models_id ) REFERENCES
MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_ARTS_FK2 FOREIGN KEY ( arts_id ) REFERENCES ARTS ( id )
ON DELETE CASCADE
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--                    IMAGES TABLE                        --
--THIS TABLE CONTAINS THE IMAGES OF ALL MODELS. IT HOLDS A REFERENCE--
--                  TO THE MODELS TABLE                   --
-------------------------------------------------------------------
CREATE TABLE IMAGES
(
 id          INTEGER NOT NULL,
 models_id     INTEGER,
 description  VARCHAR2(50 ),
 name         VARCHAR2( 30 ),
 image        ORDSYS. ORDIMAGE,
 image_thumb       ORDSYS. ORDIMAGE,
 primaryImage  CHAR(1 byte),
```

```
 CONSTRAINT IMAGE_PK PRIMARY KEY ( id )
);
-----------------------------------------------------------------


-----------------------------------------------------------------
--                    VIDEOS TABLE                          --
--THIS TABLE CONTAINS THE VIDEOS OF ALL MODELS. IT HOLDS A REFERENCE--
--                  TO THE MODELS TABLE                     --
-----------------------------------------------------------------
CREATE TABLE VIDEOS
(
 id          INTEGER NOT NULL,
 models_id     INTEGER,
 CASTING_DATE DATE, OPERATOR VARCHAR2(30 byte),
 Tape varchar2(20),
 description  VARCHAR2( 1000 ),
 title        VARCHAR2(100),
 video        ORDSYS. ORDVIDEO,
 CONSTRAINT VIDEOS_PK PRIMARY KEY ( id )
);

-----------------------------------------------------------------
--                VIDEOS_TAPES_TYPES TABLE                  --

--MANY TO MANY. IN THAT CASE WE USE A MIDDLE TABLE TO RELATE    --
--THOSE TWO. THIS TECHNIQUE WILL BE USED IN THE REST FILE AS WELL   --
-----------------------------------------------------------------
CREATE TABLE VIDEOS_TAPE_TYPES
(
 videos_id     INTEGER,
 tape_types_id INTEGER,
 CONSTRAINT VIDEOS_TAPE_TYPES_PK PRIMARY KEY ( videos_id, tape_types_id ),
 CONSTRAINT VIDEOS_TAPE_TYPES_FK1 FOREIGN KEY ( videos_id ) REFERENCES
VIDEOS ( id ) ON DELETE CASCADE,
 CONSTRAINT VIDEOS_TAPE_TYPES_FK2 FOREIGN KEY ( tape_types_id )
REFERENCES TAPE_TYPES ( id ) ON DELETE CASCADE
);
-----------------------------------------------------------------



-----------------------------------------------------------------
--                  ORDERS TABLE                        --
-----------------------------------------------------------------

CREATE TABLE ORDERS (
ID NUMBER NOT NULL,
USER_ID NUMERIC,
DESCRIPTION VARCHAR2(100),
ORDER_DATE DATE,
STAGE VARCHAR2(10),
FINAL CHAR(1 byte),
```

```
LAST_UPDATED DATE,
CONSTRAINT ORDERS_PK PRIMARY KEY(ID)
);
-------------------------------------------------------------------

-------------------------------------------------------------------
--                 ORDERS_VIDEOS TABLE                    --
-------------------------------------------------------------------

CREATE TABLE ORDERS_VIDEOS (
ORDER_ID NUMBER NOT NULL,
VIDEO_ID NOT NULL,
VORD NUMBER,
CONSTRAINT ORDERS_VIDEOS_PK PRIMARY KEY ( order_id, video_id ),
CONSTRAINT ORDERS_VIDEOS_FK1 FOREIGN KEY(ORDER_ID)
REFERENCES ORDERS(ID) ON DELETE CASCADE,
CONSTRAINT ORDERS_VIDEOS_FK2 FOREIGN KEY(VIDEO_ID)
REFERENCES VIDEOS(ID)  ON DELETE CASCADE
);

-------------------------------------------------------------------

-------------------------------------------------------------------
--                 MODELS_OTHER_CASTINGS TABLE                    --
-------------------------------------------------------------------


CREATE TABLE MODELS_OTHER_CASTINGS
(
 models_id INTEGER,
 other_castings_id INTEGER,
 order_id INTEGER,
 CONSTRAINT MODELS_OTHER_CASTINGS_UN UNIQUE (models_id,
other_castings_id,order_id),
 CONSTRAINT MODELS_OTHER_CASTINGS_FK1 FOREIGN KEY ( models_id )
REFERENCES MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_OTHER_CASTINGS_FK3 FOREIGN KEY ( order_id )
REFERENCES ORDERS ( id ) ON DELETE CASCADE
);



-------------------------------------------------------------------
--                 MODELS_JOBS TABLE                    --
-------------------------------------------------------------------
CREATE TABLE MODELS_JOBS
(
 models_id INTEGER,
 jobs_id INTEGER,
 Order_id INTEGER,
 CONSTRAINT MODELS_JOBS_UN UNIQUE (models_id, jobs_id,order_id),
```

```
 CONSTRAINT MODELS_JOBS_FK1 FOREIGN KEY ( models_id ) REFERENCES
MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_JOBS_FK2 FOREIGN KEY ( jobs_id ) REFERENCES  JOBS ( id )
ON DELETE CASCADE,
 CONSTRAINT MODELS_JOBS_FK3 FOREIGN KEY ( order_id ) REFERENCES ORDERS
( id ) ON DELETE CASCADE
);
-------------------------------------------------------------------



-----------------------------------------------------
--                    CATEGORIES TABLE                    --
-------------------------------------------------------------------

CREATE TABLE CATEGORIES
(
 id INTEGER,
 category VARCHAR2(100),
 CONSTRAINT CATEGORIES_PK PRIMARY KEY (id)
);



-------------------------------------------------------------------
--                MODELS_CATEGORIES TABLE                --
-------------------------------------------------------------------
CREATE TABLE MODELS_CATEGORIES
(
 models_id         INTEGER,
 categories_id INTEGER,
 CONSTRAINT MODELS_CATEGORIES_PK PRIMARY KEY ( models_id, categories_id ),
 CONSTRAINT MODELS_CATEGORIES_FK1 FOREIGN KEY ( models_id ) REFERENCES
MODELS ( id ) ON DELETE CASCADE,
 CONSTRAINT MODELS_CATEGORIES_FK2 FOREIGN KEY ( categories_id )
REFERENCES CATEGORIES ( id ) ON DELETE CASCADE
);

--CREATE OR REPLACE VIEW THUMBS ("MODELS_ID","MINID") AS --
--    SELECT CINE. IMAGES. MODELS_ID, --
--MIN(CINE. IMAGES. ID) as minid FROM CINE. IMAGES  --
--GROUP BY (CINE. IMAGES. MODELS_ID) WITH READ ONLY--

--CREATE OR REPLACE VIEW THUMBNAILS ("MINID","MODELS_ID",--
--    "IMAGE_THUMB") AS --
--    SELECT MINID,THUMBS. MODELS_ID,Image_Thumb --
--    from THUMBS  --
--INNER JOIN IMAGES ON MINID=IMAGES. ID WITH READ ONLY--

CREATE VIEW THUMBNAILS AS
   SELECT ID AS MINID,MODELS_ID,IMAGE_THUMB
   FROM IMAGES WHERE ID IN(SELECT MIN(ID)
```

```
FROM IMAGES GROUP BY MODELS_ID)
WITH READ ONLY;

CREATE VIEW ORDERS_VIEW AS
   select ORDERS. ID AS ID,User_ID,Login,ORDERS. description AS DESCRIPTION,
to_char(ORDER_DATE,'DD-MM-YYYY') as ORD_DATE,
Stage,MODELS_JOBS. jobs_id AS JOBS_ID,job_name,job_number,to_char(JOB_DATE,'DD-
MM-YYYY') AS JOB_DATE,
to_char(LAST_UPDATED,'DD-MM-YYYY') AS LAST_UPDATE
from CINE. ORDERS LEFT JOIN CINE. USERS
ON ORDERS. USER_ID=USERS. ID
LEFT JOIN CINE. MODELS_JOBS
ON ORDERS. ID=MODELS_JOBS. ORDER_ID
LEFT JOIN CINE. JOBS ON MODELS_JOBS. JOBS_ID=JOBS. ID WITH READ ONLY;

-------------------------------------------------
--THREE SEQUENCES FOR CONSTRACTING PRIMARY KEYS. --
--      MIGHT NOT BE USED IN THE FUTURE       --
-------------------------------------------------
CREATE SEQUENCE MODELS_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE TAPE_TYPES_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE FOREIGN_LANGUAGES_ID INCREMENT BY 1 START WITH
1000;
CREATE SEQUENCE AGENCIES_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE TEETH_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE SPORTS_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE JOBS_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE HAIR_ATTRIBUTES_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE HAIR_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE TRANSPORTATION_LICENCES_ID INCREMENT BY 1 START
WITH 1000;
CREATE SEQUENCE CHARACTER_TYPES_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE ARTS_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE ART_CATEGORIES_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE VIDEOS_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE IMAGES_ID INCREMENT BY 1 START WITH 1000;
CREATE SEQUENCE CATEGORIES_ID INCREMENT BY 1 START WITH 3;
CREATE SEQUENCE USER_CATEGORIES_ID INCREMENT BY 1 START WITH 1;
CREATE SEQUENCE USERS_ID INCREMENT BY 1 START WITH 1;
CREATE SEQUENCE ORDERS_ID INCREMENT BY 1 START WITH 1;
-------------------------------------------------


CREATE OR REPLACE DIRECTORY MODELS_VIDEOS AS 'C:\MODELS_VIDEOS';


ALTER SEQUENCE MODELS_ID MINVALUE 0;
ALTER TABLE MODELS_JOBS DROP CONSTRAINT MODELS_JOBS_UN CASCADE;
ALTER TABLE IMAGES MODIFY(DESCRIPTION VARCHAR2(100), NAME
VARCHAR2(100));
```

```
commit;

CREATE OR REPLACE TRIGGER agencies_bir
BEFORE INSERT ON AGENCIES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT agencies_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
CREATE OR REPLACE TRIGGER teeth_bir
BEFORE INSERT ON TEETH
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT teeth_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
CREATE OR REPLACE TRIGGER arts_bir
BEFORE INSERT ON ARTS
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT arts_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;

/

CREATE OR REPLACE TRIGGER art_categories_bir
BEFORE INSERT ON ART_CATEGORIES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT art_categories_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/

CREATE OR REPLACE TRIGGER CHARACTER_TYPES_bir
BEFORE INSERT ON CHARACTER_TYPES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT CHARACTER_TYPES_id. NEXTVAL
```

```
  INTO   :new. id
  FROM   dual;
END;
/


CREATE OR REPLACE TRIGGER FOREIGN_LANGUAGES_bir
BEFORE INSERT ON FOREIGN_LANGUAGES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT FOREIGN_LANGUAGES_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;

/

CREATE OR REPLACE TRIGGER HAIR_ATTRIBUTES_bir
BEFORE INSERT ON HAIR_ATTRIBUTES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT HAIR_ATTRIBUTES_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/

CREATE OR REPLACE TRIGGER HAIR_bir
BEFORE INSERT ON HAIR
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT HAIR_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/


CREATE OR REPLACE TRIGGER JOBS_bir
BEFORE INSERT ON JOBS
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT JOBS_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
```

```
CREATE OR REPLACE TRIGGER SPORTS_bir
BEFORE INSERT ON SPORTS
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT SPORTS_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;

/

CREATE OR REPLACE TRIGGER TAPE_TYPES_bir
BEFORE INSERT ON TAPE_TYPES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT TAPE_TYPES_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/

CREATE OR REPLACE TRIGGER TRANSPORTATION_LICENCES_bir
BEFORE INSERT ON TRANSPORTATION_LICENCES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT TRANSPORTATION_LICENCES_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
CREATE OR REPLACE TRIGGER CATEGORIES_bir
BEFORE INSERT ON CATEGORIES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT CATEGORIES_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
CREATE OR REPLACE TRIGGER USER_CATEGORIES_bir
BEFORE INSERT ON USER_CATEGORIES
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT USER_CATEGORIES_id. NEXTVAL
  INTO   :new. id
```

```
  FROM   dual;
END;
/
CREATE OR REPLACE TRIGGER ORDERS_bir
BEFORE INSERT ON ORDERS
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT ORDERS_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
CREATE OR REPLACE TRIGGER USERS_bir
BEFORE INSERT ON USERS
FOR EACH ROW
WHEN (new. id IS NULL)
BEGIN
  SELECT USERS_id. NEXTVAL
  INTO   :new. id
  FROM   dual;
END;
/
commit;

---------------------------------------------------------------
--       FILL USERS TABLE WITH INITIAL VALUES         --
---------------------------------------------------------------
INSERT INTO USERS VALUES ( null, 'ania','abc123',1 );
INSERT INTO USERS VALUES ( null, 'casting','bbb123',0 );


---------------------------------------------------------------
--       FILL TAPE_TYPES TABLE WITH INITIAL VALUES         --
---------------------------------------------------------------
INSERT INTO TAPE_TYPES VALUES ( null, 'DV' );
INSERT INTO TAPE_TYPES VALUES ( null, 'VHS' );
INSERT INTO TAPE_TYPES VALUES ( null, 'Hi8' );


---------------------------------------------------------------
--    FILL FOREIGN_LANGUAGES TABLE WITH INITIAL VALUES      --
---------------------------------------------------------------
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'Greek' );
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'English' );
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'French' );
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'Spanish' );
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'German' );
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'Italian' );
INSERT INTO FOREIGN_LANGUAGES VALUES ( null, 'Arabic' );
```

```
-------------------------------------------------------------
--          FILL SPORTS TABLE WITH INITIAL VALUES          --
-------------------------------------------------------------
INSERT INTO SPORTS VALUES ( null, 'Soccer' );
INSERT INTO SPORTS VALUES ( null, 'Ping Pong' );
INSERT INTO SPORTS VALUES ( null, 'Skates' );
INSERT INTO SPORTS VALUES ( null, 'Swimming' );
INSERT INTO SPORTS VALUES ( null, 'Basketaball' );
INSERT INTO SPORTS VALUES ( null, 'Handball' );
INSERT INTO SPORTS VALUES ( null, 'Tennis' );
INSERT INTO SPORTS VALUES ( null, 'Rollerblades' );
INSERT INTO SPORTS VALUES ( null, 'Fencing' );
INSERT INTO SPORTS VALUES ( null, 'Volley' );
INSERT INTO SPORTS VALUES ( null, 'Beach Volley' );
INSERT INTO SPORTS VALUES ( null, 'Baseball' );
INSERT INTO SPORTS VALUES ( null, 'Track and Field' );
INSERT INTO SPORTS VALUES ( null, 'Water Ski' );
INSERT INTO SPORTS VALUES ( null, 'Yoga' );
INSERT INTO SPORTS VALUES ( null, 'Kick Boxing' );
INSERT INTO SPORTS VALUES ( null, 'Skateboard' );
INSERT INTO SPORTS VALUES ( null, 'Windsurf' );
INSERT INTO SPORTS VALUES ( null, 'Golf' );
INSERT INTO SPORTS VALUES ( null, 'Ice Skating' );
INSERT INTO SPORTS VALUES ( null, 'Water-Polo' );
INSERT INTO SPORTS VALUES ( null, 'Bicycle' );
INSERT INTO SPORTS VALUES ( null, 'Archery' );
INSERT INTO SPORTS VALUES ( null, 'Bunji Jumping' );
INSERT INTO SPORTS VALUES ( null, 'Climbing' );
INSERT INTO SPORTS VALUES ( null, 'Snow Board' );
INSERT INTO SPORTS VALUES ( null, 'Snowski' );
INSERT INTO SPORTS VALUES ( null, 'Rafting' );
INSERT INTO SPORTS VALUES ( null, 'Rappel' );
INSERT INTO SPORTS VALUES ( null, 'Horse Back Riding' );

-------------------------------------------------------------
--       FILL  TEETH TABLE WITH INITIAL VALUES     --
-------------------------------------------------------------
INSERT INTO TEETH VALUES ( null, 'Perfect');
INSERT INTO TEETH VALUES ( null, 'Good');
INSERT INTO TEETH VALUES ( null, 'Average');
INSERT INTO TEETH VALUES ( null, 'Bad');
INSERT INTO TEETH VALUES ( null, 'Missing Teeth');

-------------------------------------------------------------
--   FILL TRANSPORTATION_LICENCES TABLE WITH INITIAL VALUES  --
-------------------------------------------------------------
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Automatic Car' );
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Gear Shift Car' );
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Speedboat' );
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Sailing' );
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Automatic Motorcycle' );
```

```
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Gearshift Motorcycle' );
INSERT INTO TRANSPORTATION_LICENCES VALUES ( null, 'Truck' );

-------------------------------------------------------------
--      FILL  HAIR_ATTRIBUTES TABLE WITH INITIAL VALUES     --
-------------------------------------------------------------
INSERT INTO HAIR_ATTRIBUTES VALUES ( 1000, 'Quality');
INSERT INTO HAIR_ATTRIBUTES VALUES ( 1001, 'Style');
INSERT INTO HAIR_ATTRIBUTES VALUES ( 1002, 'Colour');
INSERT INTO HAIR_ATTRIBUTES VALUES ( 1003, 'Length');


-------------------------------------------------------------
--           FILL  HAIR TABLE WITH INITIAL VALUES          --
-------------------------------------------------------------
INSERT INTO HAIR VALUES ( null, 'Natural', 1000 );
INSERT INTO HAIR VALUES ( null, 'Dyed', 1000 );
INSERT INTO HAIR VALUES ( null, 'Highlights', 1000 );

INSERT INTO HAIR VALUES ( null, 'Straight'      , 1001 );
INSERT INTO HAIR VALUES ( null, 'Bold'    , 1001 );
INSERT INTO HAIR VALUES ( null, 'Rasta'     , 1001 );
INSERT INTO HAIR VALUES ( null, 'Curly', 1001 );
INSERT INTO HAIR VALUES ( null, 'Frizy'      , 1001 );
INSERT INTO HAIR VALUES ( null, 'Extreme'  , 1001 );

INSERT INTO HAIR VALUES ( null, 'Blond'       , 1002 );
INSERT INTO HAIR VALUES ( null, 'Platinum'       , 1002 );
INSERT INTO HAIR VALUES ( null, 'Dark Blond'     , 1002 );
INSERT INTO HAIR VALUES ( null, 'Brown', 1002 );
INSERT INTO HAIR VALUES ( null, 'Red'     , 1002 );
INSERT INTO HAIR VALUES ( null, 'Red Blond'     , 1002 );
INSERT INTO HAIR VALUES ( null, 'Light Brown'   , 1002 );
INSERT INTO HAIR VALUES ( null, '2-Colour'     , 1002 );
INSERT INTO HAIR VALUES ( null, 'Black', 1002 );

INSERT INTO HAIR VALUES ( null, 'Short'   , 1003 );
INSERT INTO HAIR VALUES ( null, 'Medium'     , 1003 );
INSERT INTO HAIR VALUES ( null, 'Long', 1003 );

-------------------------------------------------------------
--      FILL  CHARACTER_TYPES TABLE WITH INITIAL VALUES     --
-------------------------------------------------------------
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Mom' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Dad' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Grand parent' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Child' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Funny' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Fisherman' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Business man/woman' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Elegant' );
```

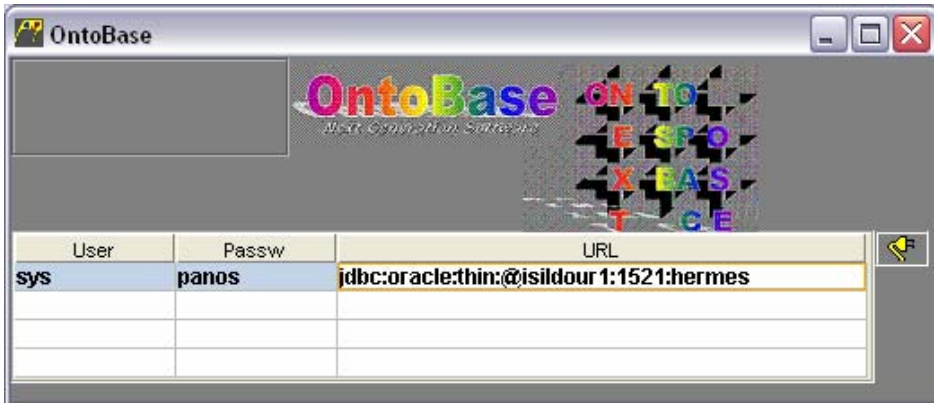
```
INSERT INTO CHARACTER_TYPES VALUES ( null, 'VIP' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Santa' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Serious' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Sporty' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Sexy' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Scientist' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Folklor' );
INSERT INTO CHARACTER_TYPES VALUES ( null, 'Military' );

-------------------------------------------------------------
--     FILL  ART_CATEGORIES TABLE WITH INITIAL VALUES       --
-------------------------------------------------------------
INSERT INTO ART_CATEGORIES VALUES ( 1000, 'Musical Instruments' );
INSERT INTO ART_CATEGORIES VALUES ( 1001, 'Circus Acts' );
INSERT INTO ART_CATEGORIES VALUES ( 1002, 'Dancing' );
INSERT INTO ART_CATEGORIES VALUES ( 1003, 'Singing' );

-------------------------------------------------------------
--          FILL ARTS TABLE WITH INITIAL VALUES          --
-------------------------------------------------------------

INSERT INTO ARTS VALUES ( null, 'Guitar', 1000 );
INSERT INTO ARTS VALUES ( null, 'Piano', 1000 );
INSERT INTO ARTS VALUES ( null, 'Violin', 1000 );

INSERT INTO ARTS VALUES ( null, 'Torches', 1001 );
INSERT INTO ARTS VALUES ( null, 'Balls', 1001 );
INSERT INTO ARTS VALUES ( null, 'Club', 1001 );
INSERT INTO ARTS VALUES ( null, 'Unicycle', 1001 );
INSERT INTO ARTS VALUES ( null, 'Cards', 1001 );
INSERT INTO ARTS VALUES ( null, 'Mime', 1001 );
INSERT INTO ARTS VALUES ( null, 'Fires', 1001 );


INSERT INTO ARTS VALUES ( null, 'Classical', 1002 );
INSERT INTO ARTS VALUES ( null, 'Modern', 1002 );
INSERT INTO ARTS VALUES ( null, 'Tap Dancing' , 1002 );

INSERT INTO ARTS VALUES ( null, 'Pop'    , 1003 );
INSERT INTO ARTS VALUES ( null, 'Folklor'  , 1003 );
INSERT INTO ARTS VALUES ( null, 'Opera'  , 1003 );
INSERT INTO ARTS VALUES ( null, 'Rock'    , 1003 );
INSERT INTO ARTS VALUES ( null, 'Hip Hop' , 1003 );

-------------------------------------------------------------
--          FILL CATEGORIES TABLE WITH INITIAL VALUES          --
-------------------------------------------------------------
INSERT INTO CATEGORIES VALUES (null, '20s' );
INSERT INTO CATEGORIES VALUES (null, '30s' );
INSERT INTO CATEGORIES VALUES (null, '40s' );
INSERT INTO CATEGORIES VALUES (null, '50s' );
```

```
INSERT INTO CATEGORIES VALUES (null, 'H20s' );
INSERT INTO CATEGORIES VALUES (null, 'H30s' );
INSERT INTO CATEGORIES VALUES (null, 'H40s' );
INSERT INTO CATEGORIES VALUES (null, 'H50s' );
INSERT INTO CATEGORIES VALUES (null, 'CH20s' );
INSERT INTO CATEGORIES VALUES (null, 'CH30s' );
INSERT INTO CATEGORIES VALUES (null, 'CH40s' );
INSERT INTO CATEGORIES VALUES (null, 'CH50s' );
INSERT INTO CATEGORIES VALUES (null, 'GR20s' );
INSERT INTO CATEGORIES VALUES (null, 'GR30s' );
INSERT INTO CATEGORIES VALUES (null, 'GR40s' );
INSERT INTO CATEGORIES VALUES (null, 'GR50s' );
INSERT INTO CATEGORIES VALUES (null, 'INTL' );
INSERT INTO CATEGORIES VALUES (null, 'LOOK ALIKES' );
INSERT INTO CATEGORIES VALUES (null, 'OLD' );
INSERT INTO CATEGORIES VALUES (null, 'BODY' );
INSERT INTO CATEGORIES VALUES (null, 'SPECIAL' );
INSERT INTO CATEGORIES VALUES (null, 'BABIES' );
INSERT INTO CATEGORIES VALUES (null, 'HANDS' );
INSERT INTO CATEGORIES VALUES (null, 'ETHNIC' );
INSERT INTO CATEGORIES VALUES (null, 'EXTREME' );
INSERT INTO CATEGORIES VALUES (null, 'BOYS/GIRLS' );
INSERT INTO CATEGORIES VALUES (null, 'DANCERS' );
INSERT INTO CATEGORIES VALUES (null, 'VIP' );
INSERT INTO CATEGORIES VALUES (null, 'TEENS' );
INSERT INTO CATEGORIES VALUES (null, 'MOUTH' );
INSERT INTO CATEGORIES VALUES (null, 'ATHLETES' );
INSERT INTO CATEGORIES VALUES (null, 'EXTRAS' );

COMMIT;
```

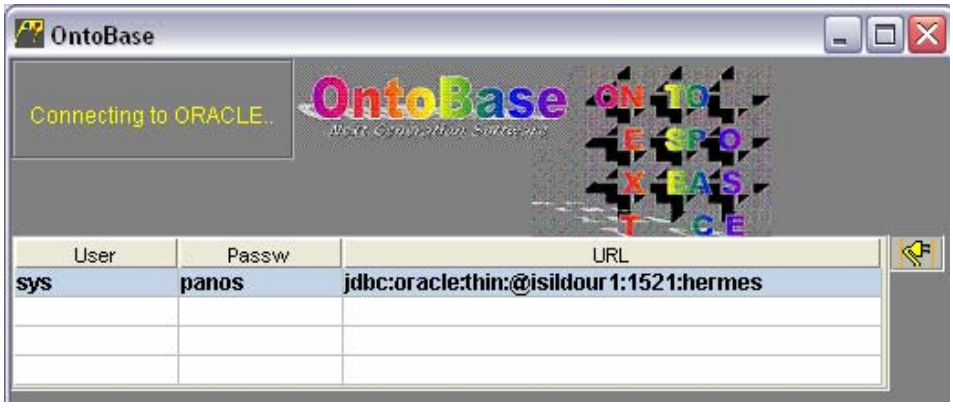
3.2 Protégé

Το επόμενο βήμα είναι η χρησιμοποίηση ενός κατεξοχήν εργαλείου για ανάπτυξη οντολογιών και για διαχείριση εφαρμογών βασισμένων στη γνώση, του Protégé 3. 0. Το εργαλείο αποτελεί μια open source προσέγγιση της ακαδημαϊκής κοινότητας του πανεπιστημίου του Stanford (http://protege. stanford. edu) βασισμένη σε Java με δυνατότητες επέκτασης μέσω plug ins και με μια πολύ ισχυρή κοινότητα από developers που περιλαμβάνει ακαδημαϊκούς, κυβερνητικά μέλη και στελέχη επιχειρήσεων και εξυπηρετεί σε εφαρμογές διαχείρησης γνώσης σε πολλούς και διαφορετικούς τομείς έρευνας.

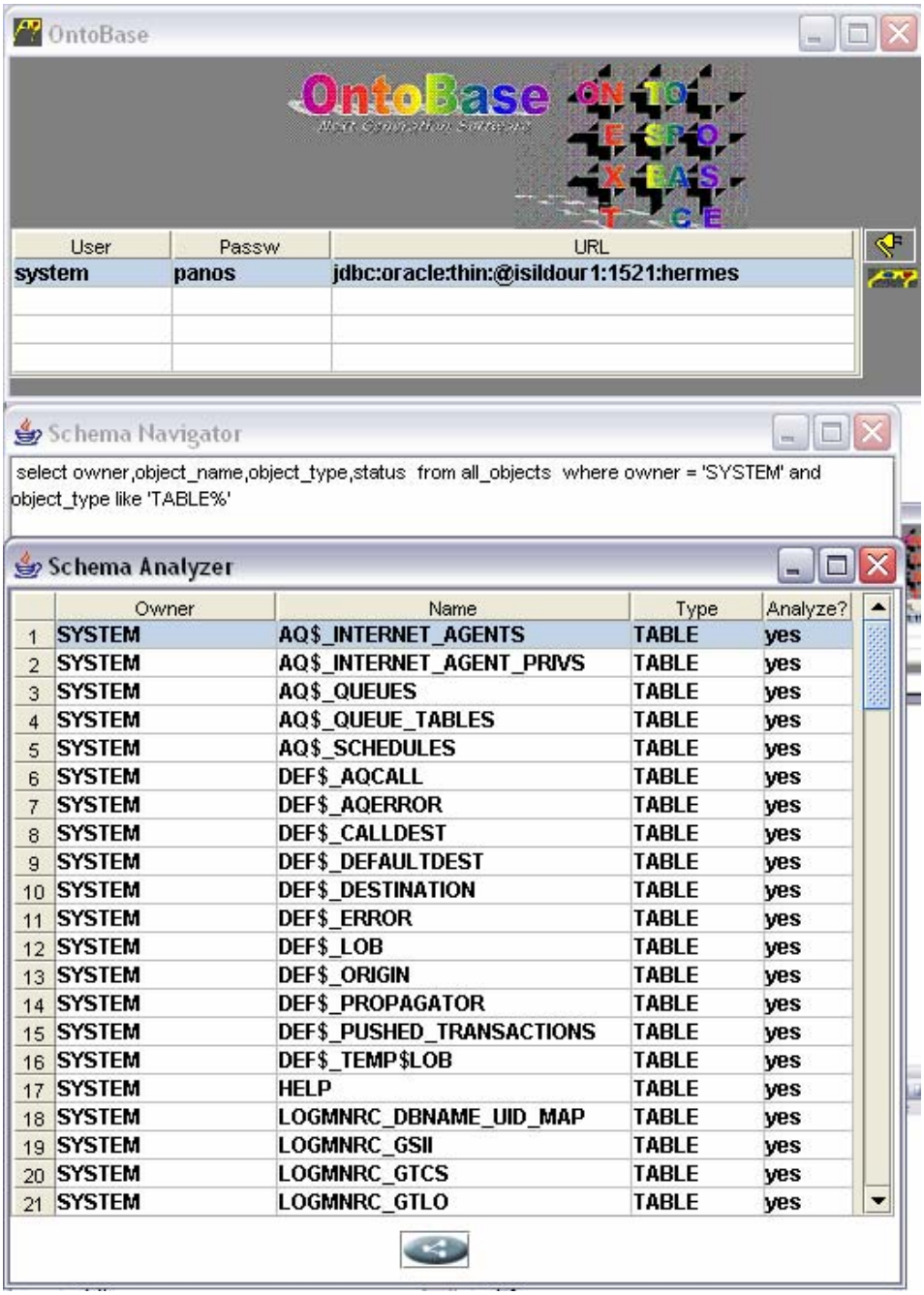
Το Protégé χρησιμοποιήθηκε αρχικά για το migration. Συγκεκριμένα έγινε χρήση του OntoBase Plug in το οποίο επιτρέπει να συνδεθείς με βάση δεδομένων Oracle και να επιλέξεις τα στοιχεία της βάσης που επιθυμεί κανείς να κάνει import. Αρχικά λοιπόν βάζουμε το url στο Onto Base, όπως φαίνεται στο σχήμα 3. 4 και κατόπιν κάνουμε connect. Με την σύνδεση στη βάση (Σχήμα 3. 5), μας προβάλλονται τα περιεχόμενα της βάσης και από εκεί επιλέγουμε ποια από αυτά θα χρησιμοποιήσουμε (Σχήμα 3. 6). Κατόπιν ξεκινά η διαδικασία του importing και τελικά λαμβάνουμε την πληροφορία της βάσης μέσα σε ένα αρχείο της μορφής. pprj και με τα υποαρχεία . pont και. pins.



Σχήμα 3. 4



Σχήμα 3. 5



Σχήμα 3. 6

Βέβαια το πιο σημαντικό στοιχείο δεν έχει αναφερθεί ακόμα. Μπορεί να είναι πολύ εύκολο να καταλάβουμε την διαδικασία της σύνδεσης στη βάση και το γεονός ότι με κάποιο τρόπο επιτυγχάνεται το import , όμως δεν είναι καθόλου σαφές το μοντέλο αντιστοίχησης μιας βάσης δεδομένων σε μία οντολογία. Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο, δεν υπάρχει μια 1-1 αντιστοιχία μεταξύ βάσης δεδομένων και οντολογίας. Ας δούμε λοιπόν ποια είναι η φιλοσοφία του migration με το συγκεκριμένο εργαλείο.

Μια βάση δεδομένων περά από τα value restrictions που επιβάλει στα πεδία της όπως int,string,date,blob,boolean και τα λοιπά παρέχει τους εξής περιορισμούς για τα tables της : Primary keys, Foreign Keys και από λειτουργικούς περιορισμούς τα triggers. Στη συγκεκριμένη περίπτωση δεν υπήρχαν triggers και περιοριστήκαμε έτσι στα primary και foreign keys.

Το αρχικά imported μοντέλο περιλαμβάνει ένα root κάτω από το οποίο αναπτύσσονται τέσσερα μεγάλα τμήματα, τα επονομαζόμενα M1,M2,M3,M4.

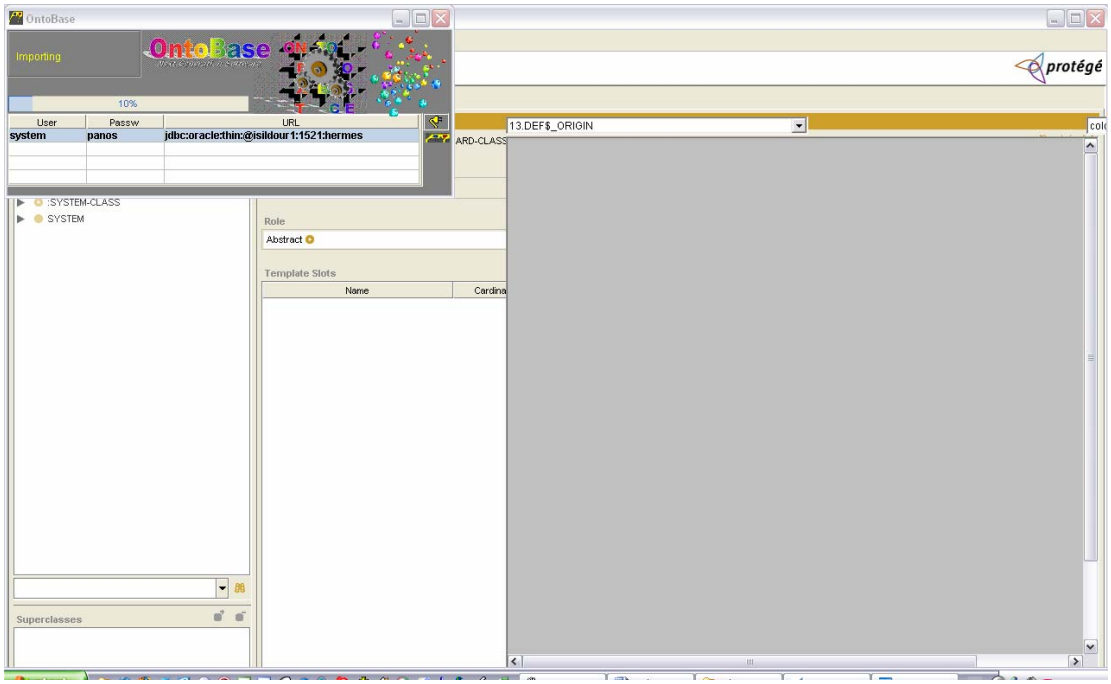
Το M1 περιλαμβάνει την βάση όπως ακριβώς θα την θέλαμε, δηλαδή κάθε table είναι μια class με όλα τα περιεχόμενα πεδία και τους περιορισμούς.

Το M2 περιλαμβάνει τα Primary Keys σαν classes,γεγονός που είναι μεν σύμφωνο με την διατήρηση της πληροφορίας αλλά που εν τέλει δεν προσθέτει κάτι επιπλέον σημασιολογικά στην οντολογία, ειδικά με τον τρόπο που παρατίθεται.

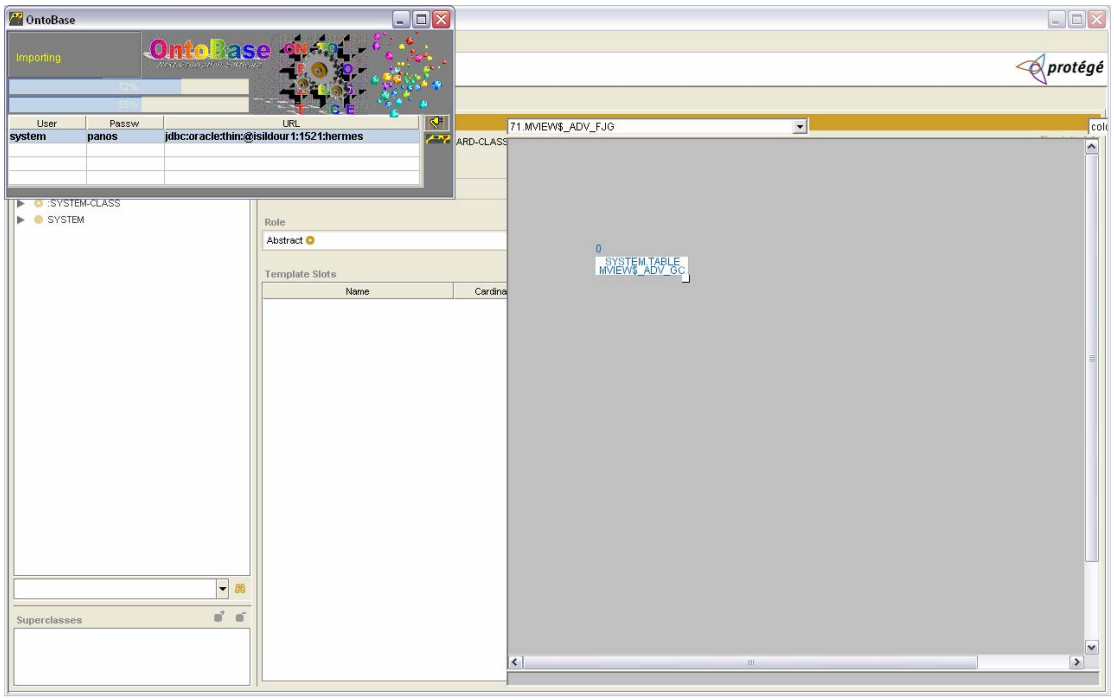
Το M3 περιλαμβάνει τα Foreign Keys σαν σχέσεις μεταξύ των κλάσεων που συνδέονται με αυτά. Η διατήρηση της πληροφορίας και πάλι επιτυγχάνεται αλλά η προσθήκη σχέσεων σε μια οντολογία είναι μια πιο ραφιναρισμένη διαδικασία και μπορεί να δώσει πολύ πιό εντυπωσιακά αποτελέσματα αν υλοποιηθεί με έναν τρόπο που να δίνει βάση στην σημασιολογία και όχι μόνο στην απλή παράθεση.

Τέλος το M4 περιλαμβάνει κάποια templates της Oracle αλλά και της σύνδεσης που επιτεύχθηκε με το OntoBase. Τα στοιχεία αυτά είναι τελείως άσχετα με το επιδιωκόμενο αποτέλεσμα και μπορούν να θεωρηθούν σαν βιβλιογραφική πληροφορία μέσα σε ένα αρχείο οντολογίας.

Με βάση τις παραπάνω πληροφορίες έγινε και η επεξεργασία της οντολογίας. Στα σχήματα 3. 7 και 3. 8 φαίνεται το αρχικό και το τελικό σχήμα της οντολογίας. Τέλος επιλέγουμε το export to owl από το Protégé και έτσι λαμβάνουμε το Final Owl όπως φαίνεται παρακάτω.

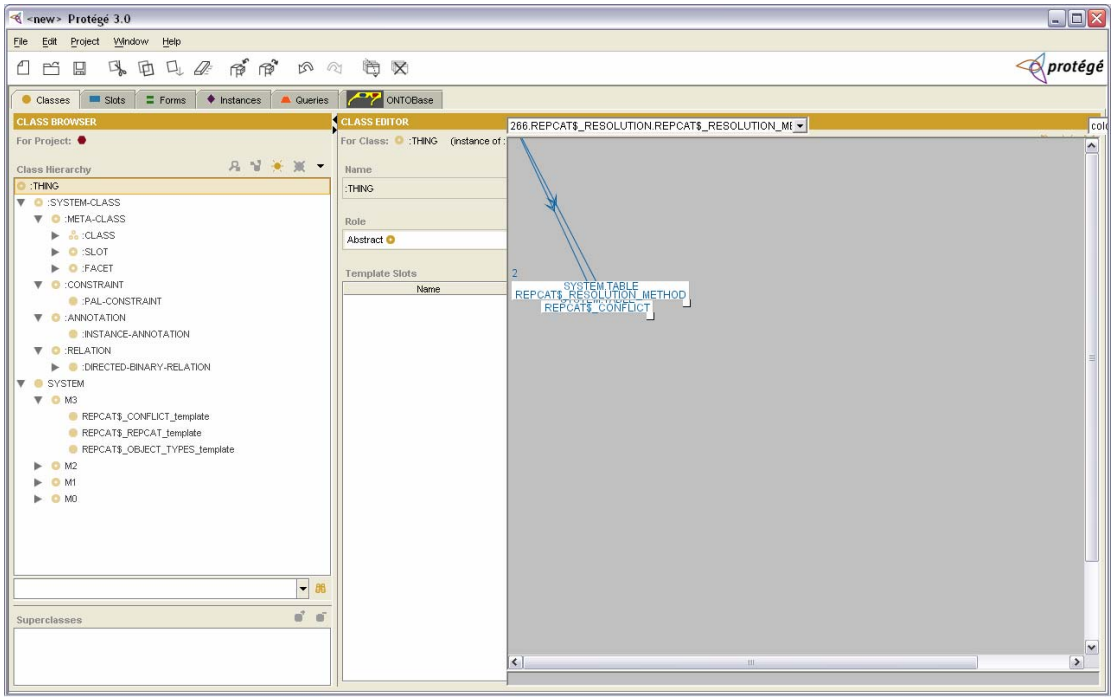


Σχήμα 3. 5



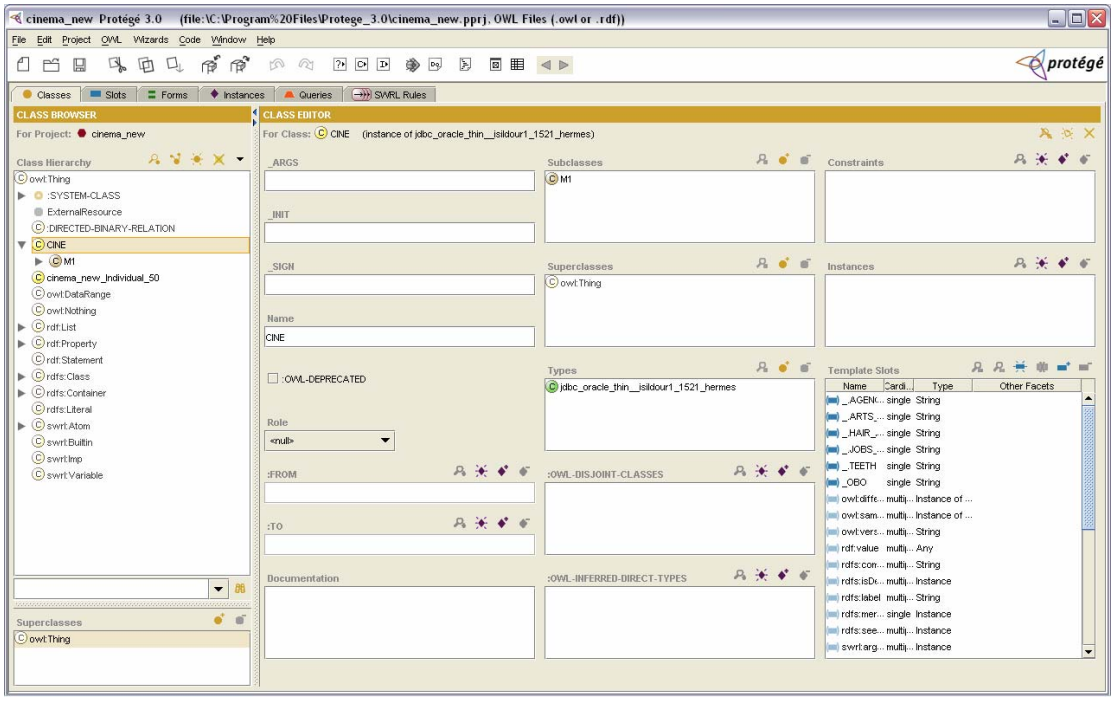
Σχήμα 3. 6

IMPORTED ONE



Σχήμα 3. 7

FINAL



Σχήμα 3. 8

FINAL OWL

```
<?xml version="1. 0"?>
<rdf:RDF
    xmlns:swrlb="http://www. w3. org/2003/11/swrlb#"
    xmlns:swrlbImport="http://www. daml. org/rules/proposal/swrlb. owl#"
    xmlns:swrl="http://www. w3. org/2003/11/swrl#"
    xmlns:protege="http://protege. stanford. edu/plugins/owl/protege#"
    xmlns:rdf="http://www. w3. org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www. w3. org/2000/01/rdf-schema#"
    xmlns:owl="http://www. w3. org/2002/07/owl#"
    xmlns="http://www. owl-ontologies. com/unnamed. owl#"
    xmlns:swrlImport="http://www. daml. org/rules/proposal/swrl. owl#"
  xml:base="http://www. owl-ontologies. com/unnamed. owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://protege. stanford.
edu/plugins/owl/protege"/>
    <owl:imports rdf:resource="http://www. daml. org/rules/proposal/swrlb.
owl"/>
    <owl:imports rdf:resource="http://www. daml. org/rules/proposal/swrl.
owl"/>
  </owl:Ontology>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. CATEGORIES">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="VIDEOS_TAPE_TYPES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MODELS. TEETH">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="MODELS"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="FOREIGN_LANGUAGES. USER_CATEGORIES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. ARTS">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="CATEGORIES. JOBS">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="HAIR">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ARTS. FOREIGN_LANGUAGES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
```

```
<owl:Class rdf:ID="HAIR. CHARACTER_TYPES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="FOREIGN_LANGUAGES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#MODELS">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="TRANSPORTATION_LICENCES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="MODELS_TRANSPORTATION_LICENCES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="HAIR. CATEGORIES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="FOREIGN_LANGUAGES. CHARACTER_TYPES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="HAIR_ATTRIBUTES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="ARTS. CHARACTER_TYPES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="MODELS_USER_CATEGORIES. MODELS">
  <rdfs:subClassOf>
    <owl:Class rdf:ID="MODELS_USER_CATEGORIES"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="MODELS_CATEGORIES">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="ARTS. JOBS">
  <rdfs:subClassOf>
    <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="ORDERS_VIDEOS">
  <rdfs:subClassOf>
```

```
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ORDERS. TAPE_TYPES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_TRANSPORTATION_LICENCES.
TRANSPORTATION_LICENCES">
    <rdfs:subClassOf rdf:resource="#MODELS_TRANSPORTATION_LICENCES"/>
  </owl:Class>
  <owl:Class rdf:ID="TEETH">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ART_CATEGORIES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="JOBS">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_SPORTS">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="HAIR. JOBS">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_OTHER_CASTINGS. ORDERS">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="MODELS_OTHER_CASTINGS"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_CHARACTER_TYPES. CHARACTER_TYPES">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="MODELS_CHARACTER_TYPES"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="CATEGORIES. USER_CATEGORIES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="CHARACTER_TYPES. USER_CATEGORIES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. FOREIGN_LANGUAGES">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
```

```
  <owl:Class rdf:ID="SPORTS">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#jdbc_oracle_thin__isildour1_1521_hermes">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="OnToSpace"/>
    </rdfs:subClassOf>
    <rdfs:label>jdbc:oracle:thin:@isildour1:1521:hermes</rdfs:label>
  </owl:Class>
  <owl:Class rdf:about="#OnToSpace">
    <rdfs:subClassOf rdf:resource="http://www. w3. org/2002/07/owl#Class"/>
  </owl:Class>
  <owl:Class rdf:ID="HAIR. ARTS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="CATEGORIES. CHARACTER_TYPES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. CHARACTER_TYPES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. HAIR">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:about="#MODELS_CHARACTER_TYPES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="CHARACTER_TYPES. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_FOREIGN_LANGUAGES. FOREIGN_LANGUAGES">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="MODELS_FOREIGN_LANGUAGES"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ORDERS_VIDEOS. ORDERS">
    <rdfs:subClassOf rdf:resource="#ORDERS_VIDEOS"/>
  </owl:Class>
  <owl:Class rdf:ID="ARTS. USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. CHARACTER_TYPES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_ARTS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="CHARACTER_TYPES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
```

```
  <owl:Class rdf:ID="ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. FOREIGN_LANGUAGES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:about="#MODELS_USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="HAIR. FOREIGN_LANGUAGES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="IMAGES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_JOBS. JOBS">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="MODELS_JOBS"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="HAIR. USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="CATEGORIES. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="JOBS. USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="ARTS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. HAIR">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. ARTS">
```

```
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="VIDEOS. MODELS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. JOBS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="HAIR. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:about="#MODELS_JOBS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="AGENCIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="ORDERS. JOBS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="TRANSPORTATION_LICENCES. USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_ARTS. ARTS">
    <rdfs:subClassOf rdf:resource="#MODELS_ARTS"/>
  </owl:Class>
  <owl:Class rdf:ID="VIDEOS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="CHARACTER_TYPES. JOBS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="USERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:about="#MODELS_FOREIGN_LANGUAGES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_HAIR">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="FOREIGN_LANGUAGES. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. JOBS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
```

```
  <owl:Class rdf:about="#MODELS_OTHER_CASTINGS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="CATEGORIES. FOREIGN_LANGUAGES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="FOREIGN_LANGUAGES. JOBS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="ARTS. CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_CATEGORIES. CATEGORIES">
    <rdfs:subClassOf rdf:resource="#MODELS_CATEGORIES"/>
  </owl:Class>
  <owl:Class rdf:ID="ORDERS. USER_CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_JOBS. ORDERS">
    <rdfs:subClassOf rdf:resource="#MODELS_JOBS"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="VIDEOS_TAPE_TYPES. VIDEOS">
    <rdfs:subClassOf rdf:resource="#VIDEOS_TAPE_TYPES"/>
  </owl:Class>
  <owl:Class rdf:ID="ARTS. ORDERS">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_SPORTS. SPORTS">
    <rdfs:subClassOf rdf:resource="#MODELS_SPORTS"/>
  </owl:Class>
  <owl:Class rdf:ID="CATEGORIES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="SPORTS. TRANSPORTATION_LICENCES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:Class rdf:ID="MODELS_HAIR. HAIR">
    <rdfs:subClassOf rdf:resource="#MODELS_HAIR"/>
  </owl:Class>
  <owl:Class rdf:ID="TAPE_TYPES">
    <rdfs:subClassOf
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
  </owl:Class>
  <owl:DatatypeProperty rdf:ID="_. SPORT">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain>
      <SPORTS rdf:ID="M1_SPORTS">
        <rdfs:subClassOf>
          <owl:DeprecatedClass rdf:ID="M1">
```

```
            <rdfs:subClassOf>
              <jdbc_oracle_thin__isildour1_1521_hermes rdf:ID="CINE"/>
            </rdfs:subClassOf>
            <rdf:type
rdf:resource="#jdbc_oracle_thin__isildour1_1521_hermes"/>
          </owl:DeprecatedClass>
        </rdfs:subClassOf>
      </SPORTS>
    </rdfs:domain>
    <rdfs:label>. SPORT</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. TITLE">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <VIDEOS rdf:ID="M1_VIDEOS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </VIDEOS>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. TITLE</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. SPORTS_ID">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. SPORTS_ID</rdfs:label>
    <rdfs:domain>
      <MODELS_SPORTS rdf:ID="M1_MODELS_SPORTS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS_SPORTS>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. TATTOOCOMMENTS">
    <rdfs:domain>
      <MODELS rdf:ID="M1_MODELS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. TATTOOCOMMENTS</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_SIGN">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>:SIGN</rdfs:label>
    <rdfs:domain rdf:resource="#OnToSpace"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. ORDER_ID">
    <rdfs:domain>
      <ORDERS_VIDEOS rdf:ID="M1_ORDERS_VIDEOS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </ORDERS_VIDEOS>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. ORDER_ID</rdfs:label>
```

```
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. HAIR_ATTRIBUTE">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. HAIR_ATTRIBUTE</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <HAIR_ATTRIBUTES rdf:ID="M1_HAIR_ATTRIBUTES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </HAIR_ATTRIBUTES>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. TRANSPORTATION_MEAN">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain>
      <TRANSPORTATION_LICENCES rdf:ID="M1_TRANSPORTATION_LICENCES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </TRANSPORTATION_LICENCES>
    </rdfs:domain>
    <rdfs:label>. TRANSPORTATION_MEAN</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. BIRTH_DATE">
    <rdfs:label>. BIRTH_DATE</rdfs:label>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. FOREIGN_LANGUAGES_ID">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <MODELS_FOREIGN_LANGUAGES rdf:ID="M1_MODELS_FOREIGN_LANGUAGES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS_FOREIGN_LANGUAGES>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. FOREIGN_LANGUAGES_ID</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. DIRECT">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. DIRECT</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. MOBILE_PHONE">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. MOBILE_PHONE</rdfs:label>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. PASSWORD">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <USERS rdf:ID="M1_USERS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </USERS>
    </rdfs:domain>
```

```
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. PASSWORD</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. TRANSPORTATION_LICENCES_ID">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. TRANSPORTATION_LICENCES_ID</rdfs:label>
    <rdfs:domain>
      <MODELS_TRANSPORTATION_LICENCES
rdf:ID="M1_MODELS_TRANSPORTATION_LICENCES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS_TRANSPORTATION_LICENCES>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. AGENCY">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. AGENCY</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. AGENCY_NAME">
    <rdfs:label>. AGENCY_NAME</rdfs:label>
    <rdfs:domain>
      <AGENCIES rdf:ID="M1_AGENCIES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </AGENCIES>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. FITNESS">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. FITNESS</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. CATEGORY">
    <rdfs:label>. CATEGORY</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <CATEGORIES rdf:ID="M1_CATEGORIES">
            <rdfs:subClassOf rdf:resource="#M1"/>
          </CATEGORIES>
          <USER_CATEGORIES rdf:ID="M1_USER_CATEGORIES">
            <rdfs:subClassOf rdf:resource="#M1"/>
          </USER_CATEGORIES>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. TAPE_TYPES_ID">
    <rdfs:domain>
      <VIDEOS_TAPE_TYPES rdf:ID="M1_VIDEOS_TAPE_TYPES">
        <rdfs:subClassOf rdf:resource="#M1"/>
```

```
      </VIDEOS_TAPE_TYPES>
    </rdfs:domain>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. TAPE_TYPES_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. FIRST_NAME">
    <rdfs:label>. FIRST_NAME</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. STAGE">
    <rdfs:domain>
      <ORDERS rdf:ID="M1_ORDERS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </ORDERS>
    </rdfs:domain>
    <rdfs:label>. STAGE</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. DRESS_NO">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. DRESS_NO</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_INIT">
    <rdfs:domain rdf:resource="#OnToSpace"/>
    <rdfs:label>:INIT</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. VIDEO">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. VIDEO</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain rdf:resource="#M1_VIDEOS"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. OPERATOR">
    <rdfs:label>. OPERATOR</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_VIDEOS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. COSTUME_NO">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. COSTUME_NO</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. CATEGORIES_ID">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
```

```
    <rdfs:label>. CATEGORIES_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain>
      <MODELS_CATEGORIES rdf:ID="M1_MODELS_CATEGORIES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS_CATEGORIES>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. ART">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <ARTS rdf:ID="M1_ARTS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </ARTS>
    </rdfs:domain>
    <rdfs:label>. ART</rdfs:label>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. ART_CATEGORY">
    <rdfs:domain>
      <ART_CATEGORIES rdf:ID="M1_ART_CATEGORIES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </ART_CATEGORIES>
    </rdfs:domain>
    <rdfs:label>. ART_CATEGORY</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. LAST_UPDATED">
    <rdfs:domain rdf:resource="#M1_ORDERS"/>
    <rdfs:label>. LAST_UPDATED</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. CASETTE_TYPE">
    <rdfs:label>. CASETTE_TYPE</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain>
      <TAPE_TYPES rdf:ID="M1_TAPE_TYPES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </TAPE_TYPES>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. PROFESSION">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:label>. PROFESSION</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. USER_CATEGORIES_ID">
    <rdfs:label>. USER_CATEGORIES_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain>
      <MODELS_USER_CATEGORIES rdf:ID="M1_MODELS_USER_CATEGORIES">
```

```
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS_USER_CATEGORIES>
    </rdfs:domain>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. FINAL">
    <rdfs:label>. FINAL</rdfs:label>
    <rdfs:domain rdf:resource="#M1_ORDERS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. JOB_DATE">
    <rdfs:label>. JOB_DATE</rdfs:label>
    <rdfs:domain>
      <JOBS rdf:ID="M1_JOBS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </JOBS>
    </rdfs:domain>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. ARTS_ARTS_FK">
    <rdfs:domain rdf:resource="#M1_ART_CATEGORIES"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. ARTS_ARTS_FK</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. HAIR">
    <rdfs:label>. HAIR</rdfs:label>
    <rdfs:domain>
      <HAIR rdf:ID="M1_HAIR">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </HAIR>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="_. MODELS_ID">
    <rdfs:label>. MODELS_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <MODELS_ARTS rdf:ID="M1_MODELS_ARTS">
            <rdfs:subClassOf rdf:resource="#M1"/>
          </MODELS_ARTS>
          <MODELS_CATEGORIES rdf:about="#M1_MODELS_CATEGORIES"/>
          <MODELS_CHARACTER_TYPES rdf:ID="M1_MODELS_CHARACTER_TYPES">
            <rdfs:subClassOf rdf:resource="#M1"/>
          </MODELS_CHARACTER_TYPES>
          <MODELS_FOREIGN_LANGUAGES
rdf:about="#M1_MODELS_FOREIGN_LANGUAGES"/>
          <MODELS_HAIR rdf:ID="M1_MODELS_HAIR">
            <rdfs:subClassOf rdf:resource="#M1"/>
          </MODELS_HAIR>
          <MODELS_USER_CATEGORIES rdf:about="#M1_MODELS_USER_CATEGORIES"/>
```

```
          <MODELS_SPORTS rdf:about="#M1_MODELS_SPORTS"/>
          <MODELS_TRANSPORTATION_LICENCES
rdf:about="#M1_MODELS_TRANSPORTATION_LICENCES"/>
          <VIDEOS rdf:about="#M1_VIDEOS"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:FunctionalProperty rdf:ID="_. CHARACTER_TYPE">
    <rdfs:domain>
      <CHARACTER_TYPES rdf:ID="M1_CHARACTER_TYPES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </CHARACTER_TYPES>
    </rdfs:domain>
    <rdfs:label>. CHARACTER_TYPE</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. PIERCING">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. PIERCING</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. VIDEO_ID">
    <rdfs:label>. VIDEO_ID</rdfs:label>
    <rdfs:domain rdf:resource="#M1_ORDERS_VIDEOS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. TEETH">
    <rdfs:label>. TEETH</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. ADDRESS">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. ADDRESS</rdfs:label>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. INPUT_DATE">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. INPUT_DATE</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. DESCRIPTION">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <MODELS rdf:about="#M1_MODELS"/>
          <ORDERS rdf:about="#M1_ORDERS"/>
          <VIDEOS rdf:about="#M1_VIDEOS"/>
        </owl:unionOf>
```

```
      </owl:Class>
    </rdfs:domain>
    <rdfs:label>. DESCRIPTION</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. SEX">
    <rdfs:label>. SEX</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. HANDS">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. HANDS</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. ARTS_ID">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. ARTS_ID</rdfs:label>
    <rdfs:domain rdf:resource="#M1_MODELS_ARTS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. ORDER_DATE">
    <rdfs:domain rdf:resource="#M1_ORDERS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. ORDER_DATE</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. LAST_NAME">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. LAST_NAME</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. CASTING_DATE">
    <rdfs:domain rdf:resource="#M1_VIDEOS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. CASTING_DATE</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. VOUCHER">
    <rdfs:label>. VOUCHER</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. JOB_NUMBER">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. JOB_NUMBER</rdfs:label>
    <rdfs:domain rdf:resource="#M1_JOBS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
```

```
  <owl:FunctionalProperty rdf:ID="_OBO">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>:OBO</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. ULEVEL">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_USERS"/>
    <rdfs:label>. ULEVEL</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. HOME_PHONE">
    <rdfs:label>. HOME_PHONE</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. USER_ID">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_ORDERS"/>
    <rdfs:label>. USER_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. SHOE_NO">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. SHOE_NO</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. PANTS_NO">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. PANTS_NO</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. JEEN_NO">
    <rdfs:label>. JEEN_NO</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. VIDEOS_ID">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. VIDEOS_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_VIDEOS_TAPE_TYPES"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. JOB_NAME">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. JOB_NAME</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_JOBS"/>
  </owl:FunctionalProperty>
```

```
  <owl:FunctionalProperty rdf:ID="_. COUNTRY">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. COUNTRY</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. HAIR_ATTRIBUTES_ID">
    <rdfs:label>. HAIR_ATTRIBUTES_ID</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. TEETH_TYPE">
    <rdfs:label>. TEETH_TYPE</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <TEETH rdf:ID="M1_TEETH">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </TEETH>
    </rdfs:domain>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. CHARACTER_TYPES_ID">
    <rdfs:label>. CHARACTER_TYPES_ID</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS_CHARACTER_TYPES"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. TATTOO">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. TATTOO</rdfs:label>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. LOGIN">
    <rdfs:domain rdf:resource="#M1_USERS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. LOGIN</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. BRA_NO">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. BRA_NO</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. HAIR_ID">
    <rdfs:domain rdf:resource="#M1_MODELS_HAIR"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. HAIR_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. BODY_TYPE">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
```

```
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. BODY_TYPE</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. LANGUAGE">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain>
      <FOREIGN_LANGUAGES rdf:ID="M1_FOREIGN_LANGUAGES">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </FOREIGN_LANGUAGES>
    </rdfs:domain>
    <rdfs:label>. LANGUAGE</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. IN_HOUSE">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdfs:label>. IN_HOUSE</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. JOBS_ID">
    <rdfs:label>. JOBS_ID</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. HEIGHT">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. HEIGHT</rdfs:label>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. ID">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <AGENCIES rdf:about="#M1_AGENCIES"/>
          <MODELS rdf:about="#M1_MODELS"/>
          <ARTS rdf:about="#M1_ARTS"/>
          <ART_CATEGORIES rdf:about="#M1_ART_CATEGORIES"/>
          <CATEGORIES rdf:about="#M1_CATEGORIES"/>
          <CHARACTER_TYPES rdf:about="#M1_CHARACTER_TYPES"/>
          <FOREIGN_LANGUAGES rdf:about="#M1_FOREIGN_LANGUAGES"/>
          <HAIR rdf:about="#M1_HAIR"/>
          <HAIR_ATTRIBUTES rdf:about="#M1_HAIR_ATTRIBUTES"/>
          <IMAGES rdf:ID="M1_IMAGES">
            <rdfs:subClassOf rdf:resource="#M1"/>
          </IMAGES>
          <JOBS rdf:about="#M1_JOBS"/>
          <TEETH rdf:about="#M1_TEETH"/>
          <ORDERS rdf:about="#M1_ORDERS"/>
          <SPORTS rdf:about="#M1_SPORTS"/>
          <TRANSPORTATION_LICENCES rdf:about="#M1_TRANSPORTATION_LICENCES"/>
          <USER_CATEGORIES rdf:about="#M1_USER_CATEGORIES"/>
          <VIDEOS rdf:about="#M1_VIDEOS"/>
          <TAPE_TYPES rdf:about="#M1_TAPE_TYPES"/>
          <USERS rdf:about="#M1_USERS"/>
        </owl:unionOf>
```

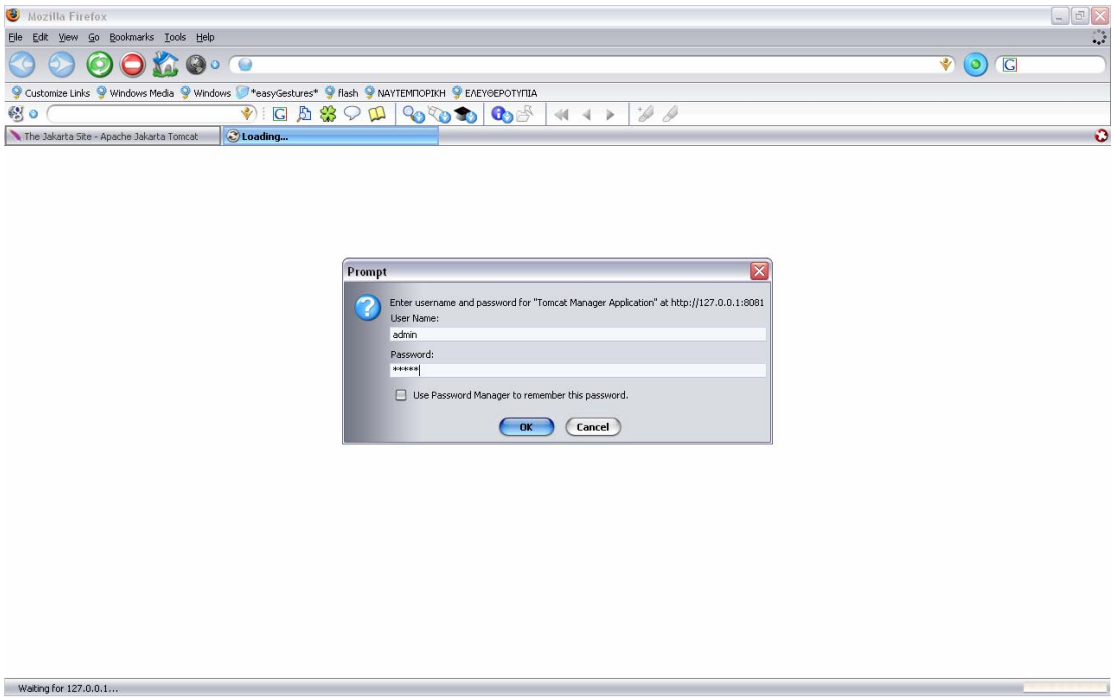
```
      </owl:Class>
    </rdfs:domain>
    <rdfs:label>. ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. CONTACT">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. CONTACT</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. EYE_COLOR">
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:label>. EYE_COLOR</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. EYES">
    <rdfs:label>. EYES</rdfs:label>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#M1_MODELS"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. OTHER_CASTINGS_ID">
    <rdfs:domain>
      <MODELS_OTHER_CASTINGS rdf:ID="M1_MODELS_OTHER_CASTINGS">
        <rdfs:subClassOf rdf:resource="#M1"/>
      </MODELS_OTHER_CASTINGS>
    </rdfs:domain>
    <rdfs:label>. OTHER_CASTINGS_ID</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. ARTS_CATEGORY_ID">
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. ARTS_CATEGORY_ID</rdfs:label>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. VORD">
    <rdfs:label>. VORD</rdfs:label>
    <rdfs:domain rdf:resource="#M1_ORDERS_VIDEOS"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_. TAPE">
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
    <rdfs:label>. TAPE</rdfs:label>
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:domain rdf:resource="#M1_VIDEOS"/>
  </owl:FunctionalProperty>
  <owl:FunctionalProperty rdf:ID="_ARGS">
    <rdfs:label>:ARGS</rdfs:label>
    <rdfs:domain rdf:resource="#OnToSpace"/>
```

```
    <rdf:type rdf:resource="http://www. w3.
org/2002/07/owl#DatatypeProperty"/>
    <rdfs:range rdf:resource="http://www. w3. org/2001/XMLSchema#string"/>
  </owl:FunctionalProperty>
  <MODELS rdf:ID="cinema_new_Individual_50"/>
  <M1_MODELS rdf:ID="Giwrgos_Gewrgiou">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
    >1. 67</_. HEIGHT>
  </M1_MODELS>
  <M1_MODELS rdf:ID="Mixalis_Mixail">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
    >1. 78</_. HEIGHT>
  </M1_MODELS>
  <M1_MODELS rdf:ID="Stavros_Stavrou">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
    >1. 55</_. HEIGHT>
  </M1_MODELS>
  <M1_MODELS rdf:ID="Vasilis_Vasileiou">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
    >1. 80</_. HEIGHT>
  </M1_MODELS>
  <MODELS_JOBS rdf:ID="M1_MODELS_JOBS">
    <rdfs:subClassOf rdf:resource="#M1"/>
  </MODELS_JOBS>
  <M1_MODELS rdf:ID="Dimitris_Dimitriou">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
    >1. 90</_. HEIGHT>
  </M1_MODELS>
  <swrl:Imp rdf:ID="Def_Is_Tall">
    <swrl:head>
      <swrl:AtomList>
        <rdf:first>
          <swrl:ClassAtom>
            <swrl:classPredicate rdf:resource="#M1_MODELS"/>
            <swrl:argument1>
              <swrl:Variable rdf:ID="x"/>
            </swrl:argument1>
          </swrl:ClassAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www. w3. org/1999/02/22-rdf-syntax-
ns#nil"/>
      </swrl:AtomList>
    </swrl:head>
    <swrl:body>
      <swrl:AtomList>
        <rdf:first>
          <swrl:ClassAtom>
            <swrl:argument1 rdf:resource="#x"/>
            <swrl:classPredicate>
              <jdbc_oracle_thin__isildour1_1521_hermes rdf:ID="Is_tall">
                <rdfs:subClassOf rdf:resource="#M1"/>
              </jdbc_oracle_thin__isildour1_1521_hermes>
            </swrl:classPredicate>
          </swrl:ClassAtom>
        </rdf:first>
        <rdf:rest rdf:resource="http://www. w3. org/1999/02/22-rdf-syntax-
ns#nil"/>
      </swrl:AtomList>
    </swrl:body>
  </swrl:Imp>
  <M1_MODELS rdf:ID="Antonis_Antoniou">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
```

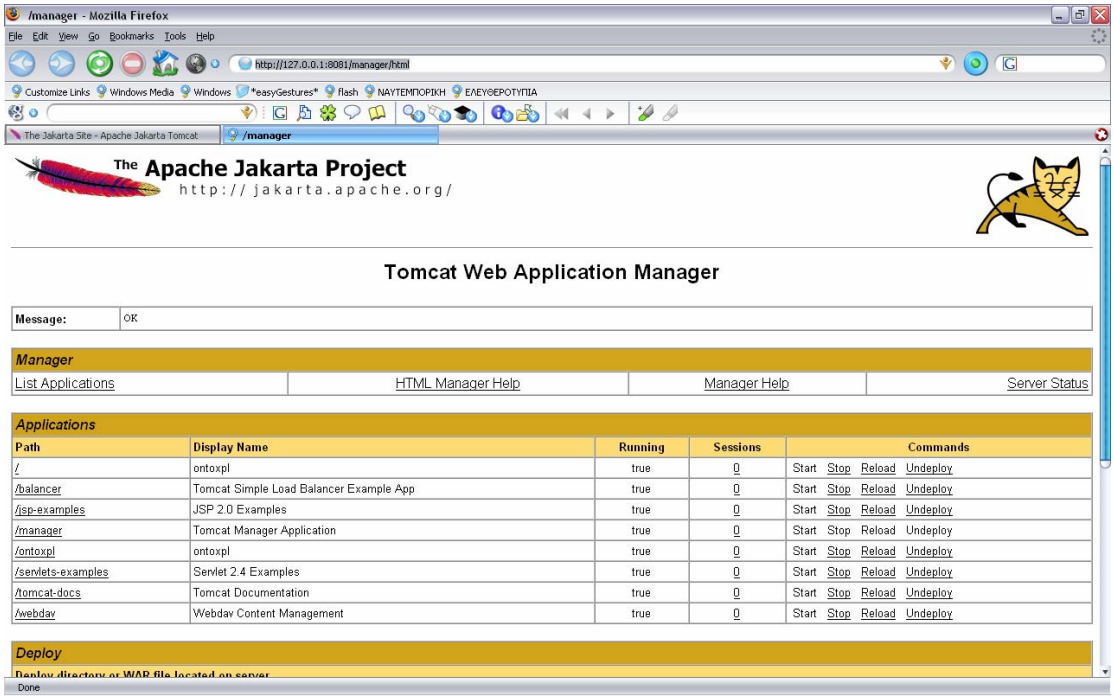
```
    >2. 01</_. HEIGHT>
  </M1_MODELS>
  <M1_MODELS rdf:ID="Giannis_Ioannou">
    <_. HEIGHT rdf:datatype="http://www. w3. org/2001/XMLSchema#string"
    >1. 82</_. HEIGHT>
  </M1_MODELS>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 1. 3, Build 225. 4)
http://protege. stanford. edu -->
```

3.3 Tomcat Web Server

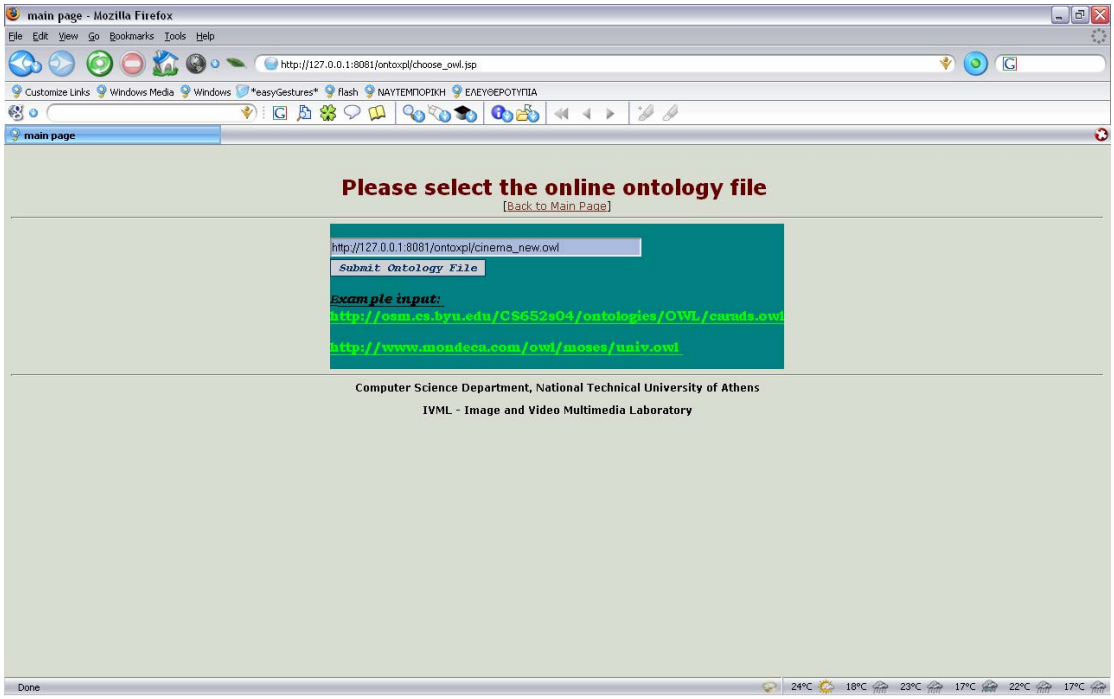
Αφού επιτεύχθηκε η δημιουργία του owl αρχείου το επόμενο βήμα ήταν να στηθεί ένας web server με υποστήριξη jsp σελίδων και προς τούτο επιλέχθηκε ο Tomcat 5. 5. 9. Κατόπιν αναπτύχθηκαν κάποιες απλές jsp σελίδες για την επιλογή ενός αρχείου. owl από ένα directory του tomcat και την φόρτωσή του στον reasoner Racer 1. 7. 24. Ο Racer προκειμένου να απαντήσει χρειάζεται ένα Dig Software Interface όπως το κλασσικό RICE και έτσι μια τέτοια υλοποίηση έγινε και στην jsp που στέλνει τα ερωτήματα σε αυτόν. Παρακάτω ακολουθούν κάποια σχήματα (3. 9 – 3. 11) για το στήσιμο του Tomcat το Tomcat Manager Console αλλά και την πρώτη σελίδα Java Server Page της εφαρμογής.



Σχήμα 3. 9



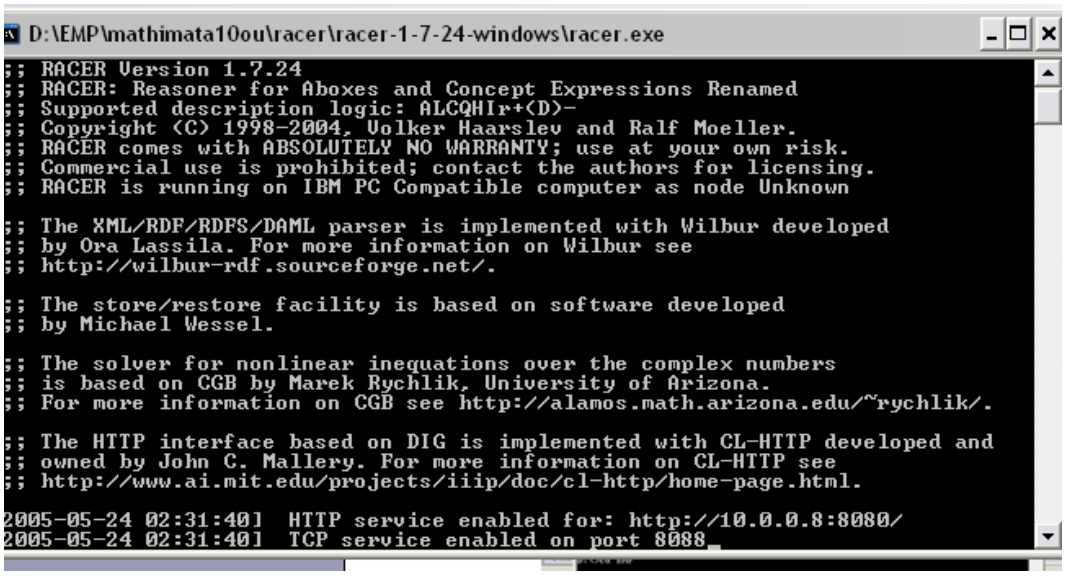
Σχήμα 3. 10



Σχήμα 3. 11

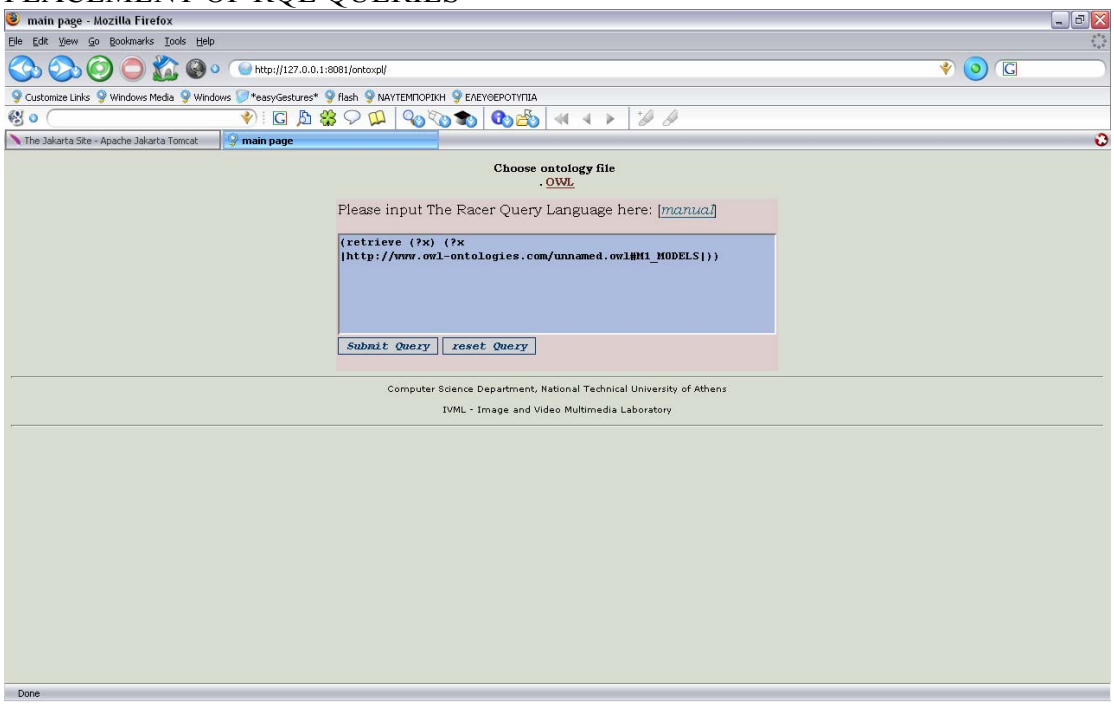
3.4 Racer, the Reasoner

Τέλος εδώ παρουσιάζεται ο Racer που πρέπει να «τρέχει» προκειμένου να απαντά τα RQL queries που τίθενται μέσω web. Η «γλώσσα» Racer Query Language είναι η δομημένη γλώσσα επερωτήσεων για τον Racer και αυτήν δέχεται μέσω του interface για να απαντήσει. Ακολουθούν κάποια χαρακτηριστικά σχήματα για το πώς τίθεται σε λειτουργία ο Racer (3. 12), πως τοποθετείται ένα sample query (3. 12) και πως επιστρέφεται η απάντηση (3. 14)



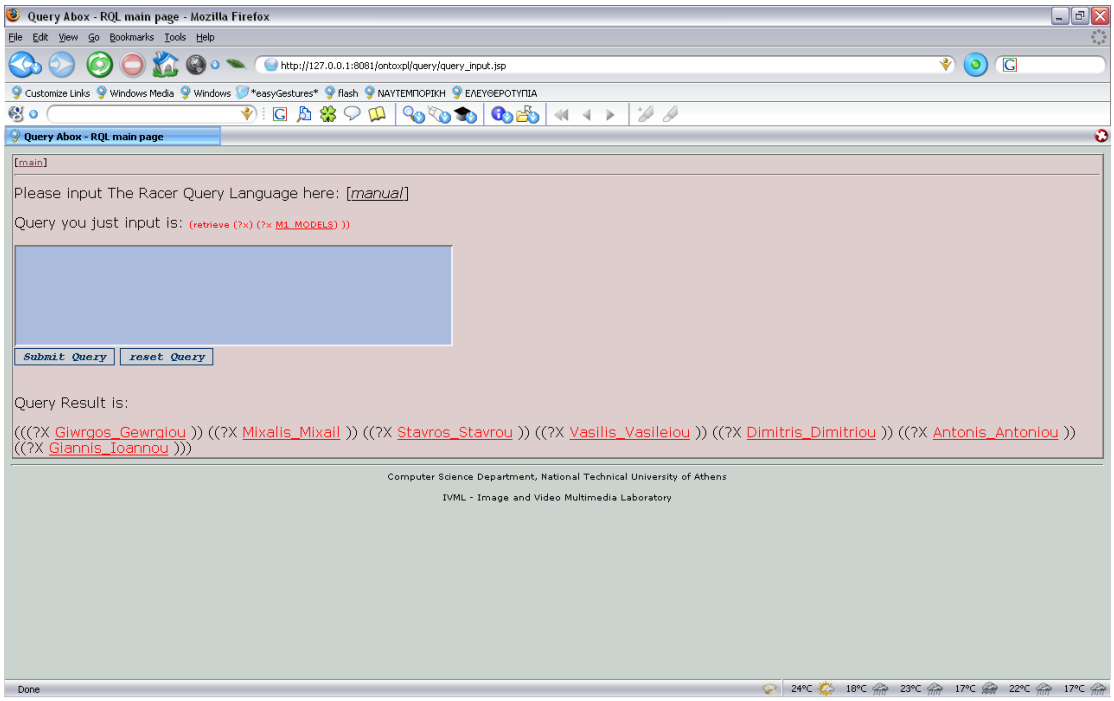
Σχήμα 3. 12

PLACEMENT OF RQL QUERIES



Σχήμα 3. 13

RACER REPLIES!!!!!



Σχήμα 3. 14

Βιβλιογραφία

- *"An Introduction to Description Logics"*, Daniele Nardi,Ronald J. Brachman

- Baader, F. , and B. Hollunder. 1991. "KRIS: Knowledge Representation and Inference System. " In *Special Issue on Implemented Knowledge Representation and Reasoning Systems*, *SIGART Bulletin* 2, no. 3 (June 1991): 108–113.

- *"Basic Description Logics"*, Franz Baader, Werner Nutt.

- Bechhofer, S. , J. Broekstra, S. Decker, M. Erdmann, D. Fensel, C. Goble, F. van Harmelen, I. Horrocks, M. Klein, D. McGuinness, E. Motta, P. Patel-Schneider, S. Staab, and R. Studer. 2000. "An Informal Description of Standard OIL and Instance OIL. " White paper. http://www. ontoknowledge. org/oil/downl/oil-whitepaper. pdf.

- Beckett, D. , ed. 2001. "RDF/XML Syntax Specification (Revised). " W3C Working Draft. http://www. w3. org/TR/2001/WD-rdf-syntax-grammar-20011218/

- Berners-Lee, T. 2002. *Weaving the Web.* San Francisco: Harper San Francisco.

- Berners-Lee, T. "*What the Semantic Web can Represent.* " http://www. w3. org/DesignIssues/RDFnot. html.

- Berners-Lee, T. , J. Hendler, and O. Lassila. 2001. "The Semantic Web. " *The ScientificAmerican*. (May 2001). http://www. scientificamerican. com/2001/0501issue/0501berners-lee. html.

- Brachman, R. J. 1978. "*A Structural Paradigm for Representing Knowledge.* " Bolt, Baranek, and Newman. BBN Report 3605. Cambridge, MA. Revision of Brachman's Ph. D. thesis. Harvard University. 1977.

- Brachman, R. J. , and H. Levesque, eds. 1985. *Readings in Knowledge Representation.* Los Altos, CA: Morgan Kaufmann.

- Brachman, R. J. , and J. G. Schmolze. 1985. "*An Overview of the KL-ONE Knowledge Representation System.* "
- Brachman R. J and Levesque H. J. 1986 . "*Expressiveness and Tractability in Knowledge Representation and reasoning*".
- DAML-ONT. http://www. daml. org/2000/10/daml-ont. html.
- DAML+OIL. 2001. http://www. daml. org/2001/03/reference. html.
- DAML-S v0. 7. http://www. daml. org/services/daml-s/0. 7/ http://www. daml. org/services/daml-s/0. 7/daml-s. html.
- Description Logics, Structural Description Logics & Modelling & Designing Knowledge Bases, Enrico Franconi franconi@cs. man. ac. uk, http://www. cs. man. ac. uk/. franconi
  Department of Computer Science, University of Manchester
- "*Description Logics for Ontologies*", Ulrike Sattler, TU Dresden
- Horrocks, I. 2002. "DAML+OIL: A Description Logic for the Semantic Web. " IEEE Intelligent Systems, 2002. Trends and Controversies.
- Horrocks, I. , and S. Tobies. 2000. Reasoning with Axioms: Theory and Practice. Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000), Breckenridge, Colorado, USA, 12-15 April 2000 (KR 2000)
- John Wiley and Sons. "*The Semantic Web, A Guide To The Future Of XML Web Services And Knowledge Management*" Michael C. Daconta, Leo J. Obrst, Kevin T. Smith,Wiley Publishing Inc. 2003.
- McGuinness, D. L. , and F. van Harmelen, eds. 2002. "*Feature Synopsis for OWL Lite and OWL*" W3C Working Draft. (July 29, 2002). http://www. w3. org/TR/2002/WD-owl-features-20020729/, http://www. w3. org/2001/sw/WebOnt/

- Noy, N. F. , R. W. Fergerson, and M. A. Musen. 2000. "*The Knowledge Model of Protege-2000: Combining Interoperability and Flexibilit.* " In *2nd International Conference on Knowledge Engineering and Knowledge Management* (EKAW '2000).

- OIL. http://www. ontoknowledge. org/oil/.

- Ontolingua: http://www. ksl. stanford. edu/software/ontolingua/.

- Ontoxpl: www. cs. concordia. ca/ying_lu/
  www. cs. concordia. ca/~haarslev/publications/DL2004-OntoXpl. pdf

- OWL: The Web Ontology Language. http://www. w3. org/2001/sw/WebOnt/.

- Patel-Schneider, P. F. , I. Horrocks, P. Hayes, and F. van Harmelen, eds. Web Ontology Language (OWL) Abstract Syntax and Semantics. W3C Working Draft. November 8, 2002.
  http://www. w3. org/TR/2002/WD-owlsemantics-20021108/
  http://www. w3. org/2001/sw/WebOnt/

- Patel-Schneider, P. F. , D. L. McGuinness, R. Brachman, L. A. Resnick, A. Borgida. 1991. "The CLASSIC Knowledge Representation System: Guiding Principles and Implementation Rationale. " In *Special Issue on Implemented Knowledge Representation and Reasoning Systems, SIGART Bulletin 2*.

- Protégé. http://protégé. stanford. edu.

- *Semantic Web Chalk Talk*, "Amateur intro to description logics", HP Corp.
  http://www. hp. com

- Smith, M. , D. L. McGuinness, R. Volz, and C. Welty, eds. 2002. "Web Ontology Language (OWL) Guide Version 1. 0. " W3C Working Draft. (November 4,2002).
  http://www. w3. org/TR/2002/WD-owl-guide-20021104/
  http://www. w3. org/2001/sw/WebOnt/

- UML. http://www. uml. org/.

- Course Notes. Available at http://www. cs. rice. edu/~vardi/sigcse/mv2. ps. gz.

- World Wide Web Consortium (W3C). http://www. w3. org/DesignIssues/
- W3C. "Resource Description Framework (RDF) Model and Syntax Specification. "
- W3C Recommendation.
- XML, RDF/S, DAML, OWL Comparison.
  http://www. daml. org/language/features. htm