Πανεπιστήμιο Κρήτης
Σχολή Θετικών Επιστημών
Τμήμα Επιστήμης Υπολογιστών

Διαλογική ενημέρωση οντοκεντρικών βάσεων δεδομένων

Δημήτρης Π. Δασκαλάκης

Μεταπτυχιακή Εργασία

Ηράκλειο, Φεβρουάριος 1996

Πανεπιστήμιο Κρήτης
Σχολή Θετικών Επιστημών
Τμήμα Επιστήμης Υπολογιστών

Διαλογική ενημέρωση οντοκεντρικών βάσεων δεδομένων

Εργασία που υποβλήθηκε από τον
ΔΗΜΗΤΡΗ Π. ΔΑΣΚΑΛΑΚΗ
ως μερική εκπλήρωση των απαιτήσεων
για την απόκτηση
ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΕΙΔΙΚΕΥΣΗΣ

Συγγραφέας:

Δημήτρης Π. Δασκαλάκης
Τμήμα Επιστήμης Υπολογιστών

Εισηγητική Επιτροπή:

Πάνος Κωνσταντόπουλος, Αναπληρωτής Καθηγητής, Επόπτης

Απόστολος Τραγανίτης, Αναπληρωτής Καθηγητής, Μέλος

Πάνος Τραχανιάς, Επίκουρος Καθηγητής, Μέλος

Δεκτή:

Πάνος Κωνσταντόπουλος
Πρόεδρος Επιτροπής Μεταπτυχιακών Σπουδών

Ηράκλειο, Φεβρουάριος 1996

Διαλογική ενημέρωση οντοκεντρικών βάσεων δεδομένων

Δημήτρης Π. Δασκαλάκης

Μεταπτυχιακή Εργασία

Τμήμα Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

ΠΕΡΙΛΗΨΗ

Οι οντοκεντρικές βάσεις δεδομένων έχουν αρκετά πολύπλοκες απαιτήσεις όσον αφορά στη φιλική εισαγωγή δεδομένων και στη διαλογική ενημέρωσή τους. Αιτία είναι οι πολλαπλές σχέσεις που μπορεί να υπάρχουν ανάμεσα στις οντότητες. Παρά τις δυσκολίες όμως, για λόγους ευχρηστίας και εργονομίας, είναι επιθυμητό να υπάρχει ένα σύστημα που να προσφέρει τη δυνατότητα εισαγωγής δεδομένων με τρόπο φιλικό στο χρήστη. Επίσης θα πρέπει αυτό το σύστημα να βοηθά και να καθοδηγεί το χρήστη σε κάθε βήμα, αλλά συγχρόνως να τον περιορίζει σε πράξεις της αρμοδιότητάς του.

Η παρούσα εργασία ασχολείται με την κατασκευή ενός διαλογικού εργαλείου για εισαγωγή πληροφορίας σε οντοκεντρικές βάσεις δεδομένων. Το εργαλείο αυτό, που ονομάζουμε Δελτίο Εισαγωγής Δεδομένων (ΔΕΔ), αποτελεί μέρος του Συστήματος Σημασιολογικού Ευρετηριασμού (ΣΣΕ). Χρησιμοποιήθηκε για την ενημέρωση βάσεων που στηρίζονται στο μοντέλο δεδομένων της γλώσσας παράστασης γνώσης Telos.

Το ΔΕΔ προσφέρει ένα πλήρες σύνολο λειτουργιών που καλύπτουν κάθε στοιχειώδη ενημέρωση του ΣΣΕ. Η υλοποίηση των λειτουργιών στο ΔΕΔ γίνεται μέσω καρτών, που δίνουν στο χρήστη τη δυνατότητα να ενημερώνει διαλογικά τις οντότητες της βάσης. Για το σκοπό αυτό έχουν κατασκευαστεί κάρτες που υλοποιούν λειτουργίες δημιουργίας, διαγραφής, μετονομασίας, ταξινόμησης, γενίκευσης και απόδοσης γνωρισμάτων για κόμβους και γνωρίσματα.

Ενα μοντέλο διεργασιών έχει χρησιμοποιηθεί, το οποίο έχει σκοπό την περιγραφή της διαδικασίας διαλογικής ενημέρωσης βάσεων δεδομένων. Το μοντέλο αυτό ακολουθεί τη διεργασιοκεντρική προσέγγιση. Η διεργασία καθορίζεται από τις οντότητες που επιτρέπεται να ενημερωθούν και τις λειτουργίες που μπορούν να γίνουν σε αυτές. Με αυτό τον τρόπο απομονώνονται τμήματα της βάσης και ενημερώνονται ανεξάρτητα από άλλα, με προκαθορισμένες λειτουργίες. Οι χρήστες μπορούν να ενημερώνουν τη βάση μέσω των διεργασιών οι οποίες τους αποδίδονται. Το ΔΕΔ, στηριζόμενο στο παραπάνω μοντέλο, χρησιμοποιεί την προσέγγιση της ενημέρωσης μέσω διεργασιών.

Το ΔΕΔ έχει τη δυνατότητα να διατηρεί ένα σύνολο από περιορισμούς ακεραιότητας, που δεν ελέγχονται από το σύστημα σημασιολογικού ελέγχου του ΣΣΕ. Αυτό επιτυγχάνεται με τη χρήση ενός μοντέλου περιορισμών που αναπτύχθηκε. Το εργαλείο προσφέρει επίσης μηχανισμούς για καθορισμό δικαιοδοσιών στους χρήστες, ενώ παρέχει μηχανισμούς καθοδήγησης και επιτάχυνσης της διαδικασίας διαλογικής ενημέρωσης.

Το ΔΕΔ μπορεί εύκολα να προσαρμόζεται στις ανάγκες κάθε εφαρμογής που καλείται να ενημερώσει, χωρίς να γίνεται χρήση γλωσσών προγραμματισμού. Αρκεί η προσαρμογή του μοντέλου διεργασιών και του μοντέλου περιορισμών στην εκάστοτε εφαρμογή. Το εργαλείο χρησιμοποιήθηκε σε ένα σύνολο εφαρμογών του ΣΣΕ με γενικά ικανοποιητικά αποτελέσματα.

Επόπτης : Πάνος Κωνσταντόπουλος
Αναπληρωτής Καθηγητής Επιστήμης Υπολογιστών
Πανεπιστήμιο Κρήτης

Interactive update of object-oriented databases

Dimitris P. Daskalakis

Master of Science Thesis

Department of Computer Science
University of Crete

ABSTRACT

Object-oriented databases have quite complicated needs concerning the user-friendly data entry and the interactive update, because of the multiple relations that may hold among the objects. In spite of the difficulties that may exist, there is a need for a system which provides the users with the capability to enter data in a friendly way. It is also desirable for this system to guide and help the user in each step and in the same time to hold the user's operations under control.

This thesis deals with the construction of an interactive tool for information acquisition to object-oriented databases. The tool, which is called Data Entry Form (DEF), is a part of the Semantic Index System (SIS). It is used for the update of databases that are based on the data model of the knowledge presentation language Telos.

The DEF offers a complete set of operations that covers the whole set of Telos primitive updates. The operations in the DEF are implemented by a set of update forms, which provides the user with the capability to update the database objects interactively. These forms are related with operations for creation, deletion, rename, instantiation, generalization and attribution for nodes and attributes.

A process model is used, in order to describe the databases interactive update process. This model is based on the task-oriented approach. A task is defined by the objects that are allowed to be updated and the operations that can be performed on these objects. In this way, parts of the database can be isolated and updated indepentendly from others, using predefined operations. The system users can update the database using the tasks which are assigned to them. The DEF, based on the prementioned model, uses the task-oriented approach.

The DEF has the ability to preserve a set of integrity constraints which are not checked by the semantic checker of the SIS system. This is accomplished by the use of an integrity constraint model which has been developed. Also the tool provides facilities for granding permission to the users. Furthermore the tool provides acceleration facilities and guiding mechanisms during the interactive update.

The DEF can easily be adapted to the needs of an application, without the use of a programming language. For this purpose, it is sufficient to adapt the process and the integrity constraint model to the application. The tool has been tested in several SIS applications, producing quite satisfactory results.

Supervisor : Panos Constantopoulos
Associate Professor of Computer Science
University of Crete

Ευχαριστίες

Ευχαριστώ τον επόπτη καθηγητή μου κ. Πάνο Κωνσταντόπουλο που μου εμπιστεύθηκε αυτή την εργασία, καθώς και για την καθοδήγηση και τις συμβουλές του.

Επίσης, όλα τα μέλη της Ομάδας Πληροφοριακών Συστημάτων και Τεχνολογίας Λογισμικού του Ινστιτούτου Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνας για τη συνεργασία και τη συμπαράστασή τους. Ιδιαίτερα το Martin Dörr, τη Μαρία Χριστοφοράκη και το Γιάννη Τζίτζικα που με τις ιδέες τους και τα σχόλιά τους, συνέβαλλαν σημαντικά στην περάτωση αυτής της εργασίας.

Ακόμα, το Ινστιτούτο Πληροφορικής για την υλικοτεχνική υποδομή και την οικονομική ενίσχυση που μου παρείχε κατά τη διάρκεια των μεταπτυχιακών μου σπουδών.

Ιδιαίτερα θα ήθελα να ευχαριστήσω τους γονείς μου και τον αδελφό μου για τη συμπαράστασή τους σε όλη τη διάρκεια των σπουδών μου.

Επίσης, τη Χριστίνα Γκριτζάπη για την πολύπλευρη συμπαράστασή της στη διάρκεια των μεταπτυχιακών μου σπουδών. Τέλος, ευχαριστώ όλους τους φίλους μου που, ο καθένας με τον τρόπο του, με βοήθησαν να συνεχίσω και να ολοκληρώσω την προσπάθειά μου.

Περιεχόμενα

Περίληψη **i**

Abstract **iii**

Ευχαριστίες **v**

1 Εισαγωγή **1**

1.1 Το πρόβλημα . . . 1

1.2 Αντικείμενο της εργασίας . . . 1

1.3 Αποτελέσματα . . . 3

1.4 Υλοποίηση . . . 3

1.5 Η οργάνωση της εργασίας . . . 4

2 Διαλογική ενημέρωση πληροφορίας **5**

2.1 Θέματα επαφής χρήσης . . . 5

2.1.1 Νοητικά και εννοιολογικά μοντέλα . . . 6

2.1.2 Ανάλυση απαιτήσεων . . . 7

2.1.3 Εισαγωγή δεδομένων . . . 8

2.2 Σχεσιακές βάσεις . . . 8

2.3 Βάσεις γνώσης . . . 9

2.4 Οντοκεντρικές βάσεις . . . 11

2.4.1 GemStone . . . 12

2.4.2 O_2 . . . 13

2.4.3 ObjectStore . . . 13

2.4.4 Ontos . . . 14

2.5 Το Σύστημα Σημασιολογικού Ευρετηριασμού . . . 14

2.5.1 Η γλώσσα Telos . . . 15

2.5.1.1 Η δομή της γλώσσας . . . 15

2.5.1.2 Δομικοί περιορισμοί . . . 16

2.5.2 Στοιχειώδεις ενημερώσεις . . . 17
2.5.3 Μαζική εισαγωγή δεδομένων . . . 18
2.5.4 Διαλογική εισαγωγή δεδομένων . . . 19
2.6 Προσεγγίσεις στη διαλογική ενημέρωση . . . 19
2.6.1 Οντοκεντρική προσέγγιση . . . 20
2.6.2 Διεργασιοκεντρική προσέγγιση . . . 20
2.7 Σύγκριση με παρόμοια συστήματα . . . 21

3 Μοντέλο Διεργασιών Ενημέρωσης 23
3.1 Η προσέγγιση . . . 23
3.2 Γενική περιγραφή μοντέλου διεργασιών . . . 24
3.3 Πραγματοποίηση μοντέλου διεργασιών . . . 25
3.3.1 Καθορισμός συνόλων . . . 26
3.4 Διεργασίες ενημέρωσης . . . 26
3.4.1 Πεδίο δράσης διεργασίας . . . 27
3.4.2 Εκχώρηση δικαιοδοσιών . . . 28
3.4.3 Καθοδήγηση του χρήστη . . . 29
3.5 Λειτουργίες ενημέρωσης . . . 29

4 Το Δελτίο Εισαγωγής Δεδομένων - Περιγραφή 35
4.1 Ροή ελέγχου . . . 36
4.2 Εκκίνηση . . . 37
4.3 Καθορισμός διεργασίας . . . 38
4.4 Καθορισμός λειτουργίας . . . 39
4.4.1 Καθορισμός λειτουργίας για κόμβους . . . 39
4.4.2 Καθορισμός λειτουργίας για γνωρίσματα . . . 41
4.5 Κάρτες ενημέρωσης . . . 41
4.5.1 Δημιουργία κόμβου . . . 43
4.5.2 Διαγραφή κόμβου . . . 43
4.5.3 Μετονομασία οντότητας . . . 44
Μετονομασία κόμβου . . . 45
Μετονομασία γνωρίσματος . . . 45
4.5.4 Ταξινόμηση οντότητας . . . 45
Ταξινόμηση κόμβου . . . 46
Ταξινόμηση γνωρίσματος . . . 47
4.5.5 Γενίκευση οντότητας . . . 48
Γενίκευση κόμβου . . . 48

Γενίκευση γνωρίσματος 48
4.5.6 Ενημέρωση γνωρισμάτων 48
Ενημέρωση γνωρισμάτων κόμβου 50
Ενημέρωση γνωρισμάτων γνωρίσματος 51
4.5.6.1 Καταστάσεις της κάρτας 51
4.5.6.2 Παρουσίαση γνωρισμάτων 53
4.5.6.3 Δημιουργία γνωρίσματος 53
4.5.6.4 Διαγραφή γνωρίσματος 55

5 Διατήρηση Περιορισμών **57**
5.1 Διατήρηση σύνδεσης ιεραρχιών 58
5.1.1 Ιεραρχίες εξειδίκευσης 58
5.1.2 Ιεραρχίες γνωρισμάτων 59
5.2 Σύνολα αναλλοίωτου πληθυσμού 60
5.3 Μη κοινά γνωρίσματα 61
5.4 Γνωρίσματα απαραίτητα για την ύπαρξη κόμβων 62
5.5 Ακολουθίες λειτουργιών 63

6 Αυτοματισμοί και διευκόλυνση του χρήστη **65**
6.1 Σύνθετες οντότητες 65
6.1.1 Ονοματοδοσία εξαρτημένων οντοτήτων 66
6.1.2 Ενημέρωση σύνθετων οντοτήτων 67
6.2 Καθορισμός πεδίου τιμών γνωρίσματος 68
6.2.1 Καθορισμός από το χρήστη 70
6.2.2 Καθορισμός από το μοντέλο 70
6.3 Επέκταση πεδίου τιμών γνωρίσματος 71

7 Εμπειρία χρήσης **75**
7.1 Εφαρμογή SIB Class Management System 75
7.2 Εφαρμογή ΚΛΕΙΩ 76
7.3 Εφαρμογή Booch 79
7.4 Εφαρμογή AAT 79
7.5 Γενικές παρατηρήσεις 80

8 Επίλογος **83**
8.1 Συμπεράσματα 83
8.2 Επεκτάσεις-Βελτιώσεις 84
8.2.1 Μοντέλο διεργασιών 84

8.2.2 Επαφή χρήσης 86
8.2.2.1 Κάρτα επιλογής διεργασίας 86
8.2.2.2 Εισαγωγή/διαγραφή χαρακτηριστικών γνωρισμάτων κλάσεων 86
8.2.3 Ερωτηματική γλώσσα 87

A Περιγραφή μοντέλου διεργασιών σε Telos **89**

Βιβλιογραφία **90**

Κατάλογος Σχημάτων

1.1 Το πλαίσιο ανάπτυξης του Δελτίου Εισαγωγής Δεδομένων 4

2.1 Καθορισμός εργαλείων απόκτησης γνώσης με μετα-εργαλεία 10
2.2 Το περιβάλλον του Συστήματος Σημασιολογικού Ευρετηριασμού 14

3.1 Σχηματική περιγραφή του Μοντέλου Διεργασιών Ενημέρωσης 24
3.2 Παράδειγμα χρήσης QueryMacro 27
3.3 Εκχώρηση δικαιοδοσιών με το μοντέλο διεργασιών 29
3.4 Καθοδήγηση του χρήστη 30

4.1 Διάγραμμα ροής ελέγχου 37
4.2 Καθορισμός διεργασιών ανά χρήστη 38
4.3 Describe Object: Καθορισμός παραμετρικής διεργασίας 39
4.4 Επιλογή λειτουργίας και κόμβου για ενημέρωση 40
4.5 Κάρτα ενημέρωσης γνωρισμάτων 42
4.6 Ενημέρωση γνωρίσματος που εκκινεί από γνώρισμα 43
4.7 Κάρτα δημιουργίας κόμβου 44
4.8 Κάρτα διαγραφής κόμβου 44
4.9 Κάρτα μετονομασίας κόμβου 45
4.10 Κάρτα μετονομασίας γνωρίσματος 46
4.11 Κάρτα ταξινόμησης κόμβου 47
4.12 Εύρεση διαθέσιμων κατηγοριών 48
4.13 Κάρτα ταξινόμησης γνωρίσματος 49
4.14 Κάρτα γενίκευσης κόμβου 50
4.15 Κάρτα γενίκευσης γνωρίσματος 51
4.16 Κάρτα ενημέρωσης γνωρισμάτων γνωρίσματος 52
4.17 Δημιουργία γνωρίσματος 54
4.18 Διαγραφή γνωρίσματος 55

5.1 Διαγραφή κόμβου από ιεραρχία εξειδίκευσης. Κατάσταση πριν τη διαγραφή. 58

5.2 Διαγραφή κόμβου από ιεραρχία εξειδίκευσης. Κατάσταση μετά τη διαγραφή. 59
5.3 Συνδεδεμένοι όροι μέσω της ‘‘ιεραρχίας’’ των ευρύτερων όρων 60
5.4 (**Facet**, *synonym*) : περίπτωση της μετα-κατηγορίας (**Individual**, *non_shared*) 61
5.5 Ακολουθία λειτουργιών καθορισμένη από το μοντέλο διεργασιών 64

6.1 Καθορισμός ονόματος εξαρτημένης οντότητας 67
6.2 Ενημέρωση σύνθετης οντότητας 68
6.3 Διαγραφή εξαρτημένης οντότητας 69
6.4 Οντότητα καθορισμού πεδίου τιμών γνωρίσματος 70
6.5 Προτεινόμενη κλάση καθορισμού τιμών γνωρίσματος 71

7.1 Ενταξη οντότητας σε πολλαπλές ιεραρχίες εξειδίκευσης 77

8.1 Χαρακτηριστικά γνωρίσματα κλάσεων 87

Κεφάλαιο 1

Εισαγωγή

1.1 Το πρόβλημα

Οι βάσεις δεδομένων έχουν αρκετά πολύπλοκες απαιτήσεις όσον αφορά στη φιλική εισαγωγή δεδομένων και στη διαλογική ενημέρωσή τους. Ειδικά στις οντοκεντρικές βάσεις δεδομένων, όπου το μοντέλο δεδομένων είναι αρκετά πλούσιο, η διαλογική ενημέρωση γίνεται ακόμα πιο δύσκολη. Αιτία είναι οι πολλαπλές σχέσεις που μπορεί να υπάρχουν ανάμεσα στις οντότητες. Ομως, για λόγους ευχρηστίας και εργονομίας, είναι επιθυμητό να υπάρχει ένα σύστημα που να προσφέρει τη δυνατότητα εισαγωγής δεδομένων με τρόπο φιλικό στο χρήστη. Για το σκοπό αυτό πρέπει αυτό το σύστημα να βοηθά και να καθοδηγεί το χρήστη σε κάθε βήμα, αλλά συγχρόνως να τον περιορίζει σε πράξεις της αρμοδιότητάς του. Ετσι, εκτός από καθοδήγηση, πρέπει να παρέχει και προστασία από τυχόν ανεπιθύμητες τροποποιήσεις σε τμήματα της βάσης. Η διατήρηση περιορισμών που επιβάλλονται από την εφαρμογή είναι μια ακόμα επιθυμητή ιδιότητα του συστήματος. Σημαντική επίσης είναι η ικανότητα προσαρμογής του στους σκοπούς της εφαρμογής που καλείται να εξυπηρετήσει, αφού έχει διαπιστωθεί πως η κατασκευή ενός γενικού εργαλείου, που χωρίς εξειδίκευση να καλύπτει όλες τις εφαρμογές, είναι ανέφικτη [EM93].

1.2 Αντικείμενο της εργασίας

Η παρούσα εργασία ασχολείται με την κατασκευή ενός διαλογικού εργαλείου για εισαγωγή πληροφορίας σε οντοκεντρικές βάσεις δεδομένων. Το εργαλείο αυτό, που ονομάζουμε Δελτίο Εισαγωγής Δεδομένων (ΔΕΔ), αποτελεί μέρος του Συστήματος Σημασιολογικού Ευρετηριασμού (ΣΣΕ). Η εφαρμογή του έγινε σε βάσεις που στηρίζονται στο μοντέλο δεδομένων της γλώσσας παράστασης γνώσης Telos.

Το ΣΣΕ έχει να παρουσιάσει ένα σύνολο στοιχειωδών πράξεων ενημέρωσης, μέσω των οποίων μπορεί κάθε οντότητα της βάσης να ενημερωθεί. Το ΔΕΔ προσφέρει ένα σύνολο λειτουργιών

που καλύπτουν κάθε στοιχειώδη ενημέρωση του ΣΣΕ, προσφέροντας έτσι πλήρη δυνατότητα ενημέρωσης της βάσης. Η υλοποίηση των λειτουργιών στο ΔΕΔ γίνεται μέσω καρτών, που δίνουν στο χρήστη τη δυνατότητα να ενημερώνει διαλογικά τις οντότητες που επιλέγει. Για το σκοπό αυτό έχουν κατασκευαστεί κάρτες που υλοποιούν λειτουργίες δημιουργίας, διαγραφής, μετονομασίας, ταξινόμησης, γενίκευσης και απόδοσης γνωρισμάτων για κόμβους και γνωρίσματα.

Καθοριστικός παράγοντας της λειτουργικότητας του εργαλείου είναι οι προδιαγραφές που παίρνει από το μοντέλο διεργασιών. Το τελευταίο είναι ένα μοντέλο που σχεδιάστηκε και υλοποιήθηκε στο ΙΠ-ΙΤΕ[1], με σκοπό την περιγραφή της διαδικασίας διαλογικής ενημέρωσης βάσεων δεδομένων. Φτάνει σε μεγάλο βαθμό ανάλυσης, ώστε τα τμήματά του είναι εύκολα υλοποιήσιμα από το ΔΕΔ. Το μοντέλο ακολουθεί τη διεργασιοκεντρική προσέγγιση, έχοντας σαν κέντρο της ενημέρωσης τη διεργασία. Η τελευταία αποδίδει το είδος της εργασίας που ο χρήστης καλείται να εκτελέσει. Η ίδια η διεργασία καθορίζει τις οντότητες που επιτρέπεται να ενημερωθούν και το είδος των λειτουργιών που μπορούν να γίνουν σε αυτές. Με βάση λοιπόν ένα τέτοιο μοντέλο, που προσαρμόζεται εύκολα στην εφαρμογή που ενημερώνουμε, ρυθμίζεται η λειτουργικότητα του ΔΕΔ[2].

Απόρροια της χρήσης του μοντέλου διεργασιών είναι ότι το ΔΕΔ χρησιμοποιεί τη διεργασιοκεντρική προσέγγιση για την ενημέρωση των βάσεων. Ετσι, ο χρήστης καλείται αρχικά να επιλέξει μια διεργασία, από ένα σύνολο προκαθορισμένων διεργασιών, μέσω των οποίων μπορεί να ενημερώσει τη βάση. Στη συνέχεια καλείται να επιλέξει λειτουργία και οντότητα προς ενημέρωση, στα πλαίσια πάντα της επιλεγμένης διεργασίας. Τέλος επιλέγει διαλογικά όσες παραμέτρους δεν καθορίζονται από το μοντέλο διεργασιών, για την επιλεγμένη λειτουργία.

Το εργαλείο έχει τη δυνατότητα να διατηρεί ένα σύνολο από περιορισμούς ακεραιότητας, με τη χρήση του μοντέλου περιορισμών που αναπτύχθηκε για αυτό το σκοπό. Ετσι διατηρούνται περιορισμοί που δεν ελέγχονται από το σύστημα σημασιολογικού ελέγχου του ΣΣΕ, όπως διατήρηση σύνδεσης ιεραρχιών εξειδίκευσης, διατήρηση συνόλων αναλλοίωτου πληθυσμού, αποφυγή απόδοσης κοινών τιμών σε γνωρίσματα για τα οποία κάτι τέτοιο επιβάλλεται, διατήρηση γνωρισμάτων απαραίτητων για την ύπαρξη κόμβων.

Το εργαλείο προσφέρει επίσης μηχανισμούς για καθορισμό δικαιοδοσιών ανάλογα με το χρήστη του συστήματος, ενώ παρέχει αυτοματισμούς, διευκολύνσεις και μηχανισμούς καθοδήγησης κατά τη διαλογική ενημέρωση. Ετσι ο χρήστης μπορεί, κατά τη δημιουργία γνωρισμάτων, να αποδίδει τιμές σε αυτά, επιλέγοντάς τις από το επιθυμητό πεδίο τιμών. Επίσης, από το σύστημα προτείνεται όνομα για τις εξαρτημένες οντότητες, ενώ ο χρήστης απλά επικυρώνει την επιλογή αυτή, αν είναι επιθυμητή. Τέλος διευκολύνεται η ενημέρωση σύνθετων οντοτήτων, με δυνατότητα ενημέρωσης των οντοτήτων που τις απαρτίζουν.

[1] Κύριος εισηγητής της κατασκευής του μοντέλου ήταν ο Martin Dörr.

[2] Η κατασκευή του ΔΕΔ και η σύνδεσή του με το μοντέλο διεργασιών έγινε από το συγγραφέα, ενώ η προσαρμογή του μοντέλου για τις εφαρμογές στις οποίες χρησιμοποιήθηκε, έγινε κυρίως από το Γιάννη Τζίτζικα.

1.3 Αποτελέσματα

Το ΔΕΔ χρησιμοποιήθηκε σε ένα σύνολο εφαρμογών του ΣΣΕ με γενικά ικανοποιητικά αποτελέσματα. Αποδείχτηκε φιλικό και εύχρηστο ιδίως στις εφαρμογές εκείνες για τις οποίες η ενημέρωση με τη διεργασιοκεντρική προσέγγιση είναι ενδεδειγμένη.

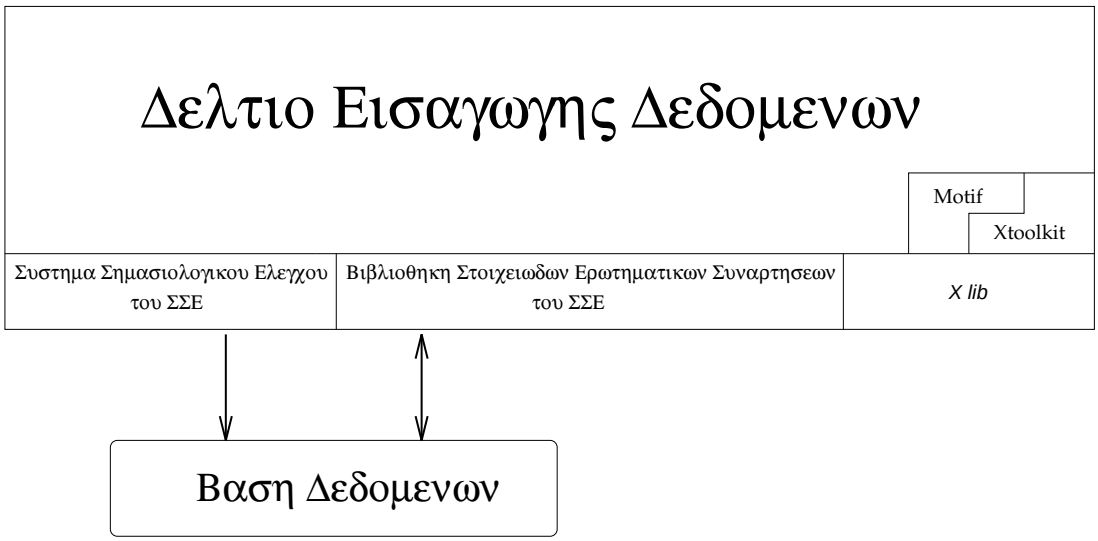
Το εργαλείο χρησιμοποιήθηκε αρχικά στην εφαρμογή SIB Class Management System. Η τελευταία έχει σκοπό την τεκμηρίωση και ανάλυση στατικών ιδιοτήτων μονάδων λογισμικού. Το ΔΕΔ σχεδιάστηκε για αυτή την εφαρμογή και φάνηκε πως οι ανάγκες για διαλογική ενημέρωσή της καλύφθηκαν πλήρως. Οντας αρκετά γενικό και εύκολα τροποποιήσιμο και εξειδικεύσιμο, το ΔΕΔ χρησιμοποιήθηκε με πολύ καλά αποτελέσματα στην εφαρμογή Booch. Σκοπός αυτής της εφαρμογής είναι να περιγράψει το περιβάλλον ανάπτυξης εφαρμογών και να δώσει στο χρήστη τη δυνατότητα να αποθηκεύσει, διαχειριστεί και εξαγάγει πληροφορία για συγκεκριμένες περιγραφές τμημάτων λογισμικού. Μια άλλη εφαρμογή στην οποία το ΔΕΔ δοκιμάστηκε με ικανοποιητικά αποτελέσματα, είναι η εφαρμογή ΑΑΤ, που έχει σκοπό την ανάπτυξη ενός πρότυπου λεξικού όρων για την τέχνη και την αρχιτεκτονική. Οι περιορισμοί ακεραιότητας που το ΔΕΔ επιβάλλει, ικανοποίησαν τις ανάγκες διατήρησης περιορισμών σε αυτήν την εφαρμογή.

Μια ακόμα χρήση του ΔΕΔ έγινε στο σύστημα ΚΛΕΙΩ, που αποτελεί ένα πληροφοριακό σύστημα μουσείων το οποίο μπορεί να χρησιμοποιηθεί σαν επιστημονικός κατάλογος μουσειακών αντικειμένων. Το μοντέλο της εφαρμογής είναι αρκετά σύνθετο σε σχέση με τα αντίστοιχα των προαναφερθεισών εφαρμογών, ενώ η προσέγγιση που ταιριάζει καλύτερα στην ενημέρωσή της είναι η οντοκεντρική[3]. Εγινε χρήση παραμετρικών διεργασιών ενημέρωσης, με σκοπό να καλύψουν ευρύ φάσμα οντοτήτων και λειτουργιών ενημέρωσης. Τα αποτελέσματα όμως δεν ήταν ιδιαίτερα ικανοποιητικά, γεγονός που εξηγείται από την προσέγγιση της ενημέρωσης που το ΔΕΔ χρησιμοποιεί, σε σχέση με αυτήν που θα ήταν πιο ενδεδειγμένη για το ΚΛΕΙΩ.

1.4 Υλοποίηση

Το ΔΕΔ υλοποιήθηκε στη γλώσσα προγραμματισμού **C++**. Η επαφή χρήσης του κατασκευάστηκε στο σύστημα παραθύρων **X** (**X Window System**) και στη βιβλιοθήκη **Motif** (**OSF Motif v1.1**). Σαν επαφή ζεύξεως με τη βάση, την οποία καλείται να ενημερώσει, έχει χρησιμοποιηθεί από τη μια η βιβλιοθήκη στοιχειωδών ερωτηματικών συναρτήσεων του ΣΣΕ και από την άλλη το σύστημα σημασιολογικού ελέγχου του ΣΣΕ. Το πλαίσιο ανάπτυξης του Δελτίου Εισαγωγής Δεδομένων φαίνεται στο σχήμα 1.1.

[3] Αντίθετα με τη διεργασιοκεντρική προσέγγιση, που έχει σαν κέντρο της ενημέρωσης μια διεργασία, η οντοκεντρική έχει σαν κέντρο της ενημέρωσης μια οντότητα, προσφέροντας λειτουργίες για την ενημέρωσή της.



Σχήμα 1.1: Το πλαίσιο ανάπτυξης του Δελτίου Εισαγωγής Δεδομένων

1.5 Η οργάνωση της εργασίας

Στο δεύτερο κεφάλαιο γίνεται ανασκόπηση της βιβλιογραφίας στο θέμα της διαλογικής ενημέρωσης πληροφορίας σε συστήματα διαχείρισης βάσεων δεδομένων, ενώ εξετάζεται με μεγαλύτερη λεπτομέρεια το ΣΣΕ.

Στο τρίτο κεφάλαιο παρουσιάζεται το μοντέλο διεργασιών ενημέρωσης.

Στο τέταρτο κεφάλαιο παρουσιάζεται το ΔΕΔ με παραδείγματα χρήσης του από εφαρμογές του ΣΣΕ.

Στο πέμπτο κεφάλαιο γίνεται παρουσίαση του μοντέλου περιορισμών στο οποίο στηρίζεται το ΔΕΔ για τη διατήρηση περιορισμών ακεραιότητας της βάσης.

Στο έκτο κεφάλαιο παρουσιάζονται αυτοματισμοί και διευκολύνσεις που το ΔΕΔ προσφέρει κατά τη διάρκεια της διαλογικής ενημέρωσης.

Στο έβδομο κεφάλαιο εξετάζονται οι ιδιαιτερότητες των εφαρμογών στις οποίες το ΔΕΔ χρησιμοποιήθηκε και παρουσιάζονται οι λόγοι που θεωρούμε ότι συνετέλεσαν στην επιτυχή ή όχι χρήση του ΔΕΔ σε αυτές.

Στο όγδοο κεφάλαιο παρουσιάζονται τα συμπεράσματα της εργασίας, ενώ παραθέτονται προτάσεις για επεκτάσεις και βελτιώσεις.

Κεφάλαιο 2

Διαλογική ενημέρωση πληροφορίας

Στο κεφάλαιο αυτό εξετάζουμε τους τρόπους με τους οποίους γίνεται ενημέρωση πληροφορίας στα συστήματα διαχείρισης βάσεων δεδομένων (ΣΔΒΔ). Ενδιαφερόμαστε ιδιαίτερα για τη διαλογική ενημέρωση της πληροφορίας. Αρχικά εξετάζουμε θέματα σχετικά με την επαφή χρήσης συστημάτων, ιδιαίτερα όσον αφορά στην εισαγωγή δεδομένων. Στη συνέχεια ερευνούμε το θέμα της διαλογικής ενημέρωσης στα σχεσιακά συστήματα και τα συστήματα βάσεων γνώσης. Σημαντικό σημείο που εξετάζουμε είναι η ευκολία προσαρμογής του εργαλείου διαλογικής ενημέρωσης στις ανάγκες της εκάστοτε εφαρμογής. Μας ενδιαφέρουν ιδιαίτερα τα οντοκεντρικά συστήματα και δίνουμε περισσότερη έμφαση σε αυτά. Κάνουμε μεγαλύτερη ανάλυση στο Σύστημα Σημασιολογικού Ευρετηριασμού, που αποτελεί το περιβάλλον ανάπτυξης της παρούσας εργασίας. Τέλος παρουσιάζουμε δύο είδη προσεγγίσεων που θεωρούμε ότι τα συστήματα ενημέρωσης δεδομένων ακολουθούν, για την επίτευξη του σκοπού τους.

2.1 Θέματα επαφής χρήσης

Η διαλογική ενημέρωση εμπεριέχει την έννοια της αλληλεπίδρασης του χρήστη με το σύστημα. Από τη σχετική ερευνητική περιοχή μαθαίνουμε ότι η αλληλεπίδραση ανθρώπου-υπολογιστή, μπορεί να θεωρηθεί η τέχνη της κατασκευής συστημάτων επικοινωνίας, που σκοπό έχουν τη μεταφορά χρήσιμης πληροφορίας από τον άνθρωπο στον υπολογιστή και αντίστροφα [Mad90]. Τα συστήματα αυτά ονομάζουμε επαφές χρήσης (user interfaces). Για να επιτευχθεί ικανοποιητική συμπεριφορά ενός συστήματος επαφής χρήσης, πολλές φορές χρειάζεται να τροποποιηθεί η συμπεριφορά του χρήστη απέναντι σε αυτό, ενώ απαραίτητη είναι επίσης η τροποποίηση της ‘‘συμπεριφοράς’’ του συστήματος. Συχνά ο σχεδιαστής του συστήματος το επανασχεδιάζει προκειμένου να ικανοποιήσει τις ανάγκες του χρήστη.

2.1.1 Νοητικά και εννοιολογικά μοντέλα

Το υψηλότερο στάδιο στο σχεδιασμό ενός συστήματος επαφής χρήσης, είναι η κατασκευή του *εννοιολογικού μοντέλου (conceptual model).* Σύμφωνα με το [May92], εννοιολογικό μοντέλο είναι το γενικότερο εννοιολογικό πλαίσιο, βάση του οποίου ο σχεδιαστής του συστήματος παρουσιάζει και εξηγεί τη συμπεριφορά του. Για παράδειγμα, ένα πολύ πετυχημένο εννοιολογικό μοντέλο στην περιοχή των υπολογιστών, είναι ο παραλληλισμός συστημάτων επαφής χρήσης με το περιβάλλον εργασίας ενός γραφείου. Εμφανίστηκε αρχικά στον Macintosh της Apple, με την παρουσίαση συστημάτων επαφής χρήσης που δίνουν στο χρήστη την αίσθηση ότι δουλεύει σε ένα γραφείο. Η ίδια η οθόνη του υπολογιστή μοιάζει με γραφείο, με οικεία αντικείμενα πάνω και γύρω του, όπως κείμενα, ντοσιέ, ρολόι, καλάθι αχρήστων κ.α.

Στενά συνδεδεμένο με το εννοιολογικό είναι το *νοητικό μοντέλο (mental model)*, που ορίζεται σαν η εσωτερική αναπαράσταση της τρέχουσας αντίληψης και κατανόησης ενός συστήματος από το χρήστη. Το εννοιολογικό μοντέλο μπορεί να θεωρηθεί η προσπάθεια του σχεδιαστή, να διευκολύνει τους χρήστες στην ανάπτυξη ενός κατάλληλου νοητικού μοντέλου.

Για να δώσουμε ένα απλό παράδειγμα νοητικού μοντέλου, εκτός της περιοχής των υπολογιστών αυτή τη φορά, μπορούμε να φανταστούμε πώς ο μέσος αυτοκινητιστής αντιλαμβάνεται τη λειτουργία της μηχανής του αυτοκινήτου. Χρησιμοποιεί ένα απλό νοητικό μοντέλο, που περιλαμβάνει μερικά βασικά τμήματά του, όπως δοχείο βενζίνης, μπαταρία, κλειδί εκκίνησης, ταχύτητες, δοχείο λαδιού, μπουζί. Το μοντέλο περιλαμβάνει γνώσεις για απλές σχέσεις των τμημάτων αυτών με τη συμπεριφορά του αυτοκινήτου.

Η ανάπτυξη ενός νοητικού μοντέλου από μέρους του χρήστη είναι ιδιαίτερα σημαντική. Σύμφωνα με τα [May92], [Nor]:

1. Ενα νοητικό μοντέλο περιέχει υποθέσεις για τα μη ορατά μέρη του συστήματος. Βάσει αυτού οι άνθρωποι συνδέουν τα ορατά μέρη του συστήματος, με τις αφανείς λειτουργίες του.

2. Οι άνθρωποι τείνουν να λύνουν καινούργια προβλήματα, κάνοντας χρήση υπαρχόντων νοητικών μοντέλων, υποθέτοντας αρχικά μέγιστη ομοιότητα μεταξύ του προβλήματος και του μοντέλου. Κάνουν περισσότερα λάθη όταν αναφορές σε υπάρχοντα νοητικά μοντέλα είναι αδύνατες ή μη προφανείς.

3. Οι άνθρωποι τείνουν να βασίζονται στην άμεση απόκριση του συστήματος, παρά στη χρήση λογικής, προκειμένου να εκτιμήσουν την ορθότητα του νοητικού τους μοντέλου. Ετσι χωρίς να σκέφτονται λογικά πολλές φορές, περιμένουν την άμεση και εμφανή αντίδραση του συστήματος, ώστε να τροποποιήσουν το τυχόν ακατάλληλο νοητικό τους μοντέλο.

4. Πολλά λάθη στη χρήση συστημάτων, μπορούν να αποδοθούν σε συμπεράσματα που εξάγονται από λανθασμένα ή ακατάλληλα νοητικά μοντέλα.

5. Συχνά οι άνθρωποι προτιμούν να κάνουν περιττές φυσικές κινήσεις, όταν επικοινωνούν με ένα σύστημα, παρά να σχεδιάζουν λογικά τις ενέργειές τους. Αυτό είναι ιδίως αληθές όταν ένας απλοποιημένος κανόνας επικοινωνίας εφαρμόζεται σε ένα σύνολο συστημάτων. Κάτι τέτοιο βοηθά στην αποφυγή συγχύσεων.

2.1.2 Ανάλυση απαιτήσεων

Ενας πρωταρχικός παράγοντας στο σχεδιασμό συστημάτων επαφής χρήσης, είναι η γνώση του χρήστη που θα το χρησιμοποιήσει. Ενας ειδικός χρήστης, που κάνει συχνή χρήση ενός συστήματος, έχει τελείως διαφορετικές απαιτήσεις από αυτό, σε σχέση με τις απαιτήσεις ενός χρήστη με λιγότερο ειδικές γνώσεις, ή ακόμα αυτές ενός αρχάριου. Δεν υπάρχει σύστημα επαφής χρήσης που να ικανοποιεί όλους τους χρήστες. Αυτός είναι ο λόγος που, πριν το σχεδιασμό του, πρέπει να ξέρουμε σε ποιους απευθύνεται και ποιες καταστάσεις θα κληθεί να εξυπηρετήσει. Σύμφωνα με το [Shn92], ‘‘το να γνωρίζεις το χρήστη’’ (‘‘know the user’’) είναι μια βασική αρχή, που όμως συχνά δεν εκτιμάται όσο θα έπρεπε.

Εχοντας αναλύσει τις ανάγκες του χρήστη, ο σχεδιαστής πρέπει να καθορίσει τους στόχους που το σύστημα καλείται να πραγματοποιήσει. Η επιτυχής ανάλυση των στόχων έχει καθοριστική σημασία για το σωστό σχεδιασμό του συστήματος, αν και συχνά νέοι στόχοι εμφανίζονται εκ των υστέρων.

Εχοντας αποφασίσει για τα παραπάνω, ο σχεδιαστής καθορίζει το είδος της αλληλεπίδρασης, τον τρόπο δηλαδή με τον οποίο θα επικοινωνεί ο χρήστης με το σύστημα. Υπάρχουν συγκεκριμένα είδη αλληλεπίδρασης που ένα σύστημα επαφής χρήσης μπορεί να χρησιμοποιήσει [Shn92], [May92], [Dix93]:

- επιλογή από μενού: ο χρήστης επιλέγει από λίστα διαθέσιμων το επιθυμητό.
- συμπλήρωση φόρμας: συχνά χρησιμοποιείται όταν είναι επιθυμητή η εισαγωγή δεδομένων. Ο χρήστης καλείται να εισάγει πληροφορία σε κενά που παρουσιάζονται σε κατάλληλα διαμορφωμένες φόρμες.
- γλώσσα εντολών: ο χρήστης μαθαίνει το συντακτικό μιας γλώσσας και μέσω αυτής δίνει εντολές στο σύστημα.
- ερώτηση/απόκριση: το σύστημα θέτει ερωτήσεις, στις οποίες καλείται να απαντήσει ο χρήστης.
- φυσική γλώσσα: ο χρήστης επικοινωνεί με το σύστημα σε φυσική γλώσσα.
- άμεσος χειρισμός: ο σχεδιαστής του συστήματος δημιουργεί μια οπτική αναπαράσταση του πεδίου δράσης, ενώ ο χρήστης μπορεί να χειριστεί άμεσα τα αντικείμενα αυτού του πεδίου.

Υπάρχουν πλεονεκτήματα και μειονεκτήματα σε κάθε προσέγγιση [Shn92], [May92] και η επιλογή γίνεται ύστερα από στάθμισή τους. Καθοριστικός παράγοντας στην επιλογή είναι το είδος του χρήστη, για τον οποίον το σύστημα προορίζεται. Συχνή είναι η κατασκευή συστημάτων που χρησιμοποιούν περισσότερες από μία προσεγγίσεις.

2.1.3 Εισαγωγή δεδομένων

Σύμφωνα με το [Shn92] υπάρχει ένα σύνολο κανόνων, οι οποίοι καλό είναι να ακολουθούνται από ένα σύστημα επαφής χρήσης, που ασχολείται με το έργο της εισαγωγής δεδομένων:

- συνέπεια ενεργειών εισαγωγής δεδομένων: παρόμοιες ακολουθίες ενεργειών για εισαγωγή δεδομένων πρέπει να ακολουθούνται σε όλες τις περιπτώσεις. Παρεκκλίσεις από το γενικό κανόνα ξαφνιάζουν δυσάρεστα το χρήστη.

- ελαχιστοποίηση ενεργειών από μέρους του χρήστη: λιγότερες ενέργειες από μέρους του χρήστη σημαίνει μεγαλύτερη παραγωγικότητα και συνήθως λιγότερα λάθη. Αντικαθιστώντας την πληκτρολόγηση χαρακτήρων με ένα απλό πάτημα πλήκτρου, ή με επιλογή από ένα μενού με το ποντίκι, δομείται το έργο της απόφασης επιλογής και εξαλείφεται η πιθανότητα για τυπογραφικά λάθη. Περιττή εισαγωγή δεδομένων, πρέπει να αποφευχθεί, αφού είναι ενοχλητικό για το χρήστη να εισάγει την ίδια πληροφορία σε διαφορετικά σημεία, ενώ αυξάνει την πιθανότητα για λάθη.

- ελαχιστοποίηση φορτίου μνήμης στο χρήστη: δεν πρέπει το σύστημα να απαιτεί από το χρήστη την απομνημόνευση κωδικών και συντακτικά σύνθετων εντολών.

- συμβατότητα εισαγωγής και παρουσίασης δεδομένων: η διάταξη των δεδομένων όπως αυτά παρουσιάζονται, πρέπει να είναι όσο το δυνατό πιο κοντά στη διάταξη με την οποία ο χρήστης καλείται να τα εισάγει. Ετσι αποφεύγεται πρόκληση σύγχυσης στο χρήστη.

- ευελιξία για έλεγχο από το χρήστη στην εισαγωγή δεδομένων: οι έμπειροι χρήστες ίσως προτιμούν την εισαγωγή δεδομένων με μια ακολουθία που μπορούν να ελέγξουν. Είναι καλό να δίνεται η δυνατότητα σε αυτούς να καθορίζουν τη σειρά με την οποία γίνεται η εισαγωγή δεδομένων. Πρέπει όμως αυτό να γίνεται με μεγάλη προσοχή, αφού μπορεί η ευελιξία να συγκρούεται με τη συνέπεια στην εισαγωγή δεδομένων.

2.2 Σχεσιακές βάσεις

Οι σχεσιακές βάσεις δεδομένων προσφέρουν ώριμη τεχνολογία με ένα σταθερό και κοινά αποδεκτό μοντέλο δεδομένων και τη χρήση της SQL ως γλώσσας διαχείρισης των δεδομένων. Το σχήμα της βάσης είναι απλό, συγκρινόμενο με το αντίστοιχο των οντοκεντρικών βάσεων, και υπάρχει σαφής

διαχωρισμός του από τα δεδομένα. Αλλαγές στα δεδομένα είναι συχνές ενώ αλλαγές στο σχήμα συμβαίνουν σπάνια.

Το χαρακτηριστικό σε αυτά τα συστήματα είναι ότι η διαλογική ενημέρωση των δεδομένων τους, γίνεται με τη βοήθεια δελτίων (Forms Based Interface). Για τον καθορισμό της εμφάνισής τους αναπτύσσονται προγράμματα γραμμένα σε ειδική γλώσσα, διαφορετική για κάθε ΣΔΒΔ [Rod90]. Οι συνήθεις τύποι επεξεργασίας, αναζήτηση (search), εισαγωγή (insert), διαγραφή (delete) και ενημέρωση (update), συνδέονται με επιλογές των δελτίων. Μια εφαρμογή σε ένα τυπικό σχεσιακό σύστημα περιέχει ένα σύνολο δελτίων μέσω των οποίων μπορεί ο χρήστης να ερευνήσει τα περιεχόμενα της βάσης. Μέσω αυτών των δελτίων επίσης, εισάγει νέα δεδομένα ή τροποποιεί ήδη υπάρχοντα.

Αναφέρουμε ενδεικτικά, γνωστά σχεσιακά συστήματα και δυνατότητες που αυτά παρέχουν στον τομέα της ενημέρωσης δεδομένων.

Η **Ingres** με το Vifred, ένα εργαλείο για δημιουργία δελτίων, δίνει τη δυνατότητα στο χρήστη να καθορίσει την εμφάνιση του δελτίου, να συσχετίσει πεδία του με πεδία της βάσης και να καθορίσει κριτήρια εγκυρότητας για τις τιμές που δέχεται ένα πεδίο.

Η **Oracle** με το Easy∗SQL και SQL∗QMX, δίνει δυνατότητα δημιουργίας παρόμοιων δελτίων. Επιλέγοντας επαφή χρήσης τύπου QueryByExample, δίνει τη δυνατότητα, ανάμεσα στα άλλα, να σχεδιάσει κανείς έναν πίνακα στην οθόνη. Είναι μια επαφή χρήσης, ικανή να εκμεταλλευτεί τις περισσότερες ευκολίες της γλώσσας SQL. Το κόστος βέβαια είναι ότι γίνεται σχετικά δύσχρηστη από χρήστες με μικρή εξοικείωση με την SQL.

Παρόμοιες δυνατότητες δίνουν η **Sybase** με το APT∗SQL, η **Informix** με το Perform και το Informix-4GL και η **Accell** με το Paint

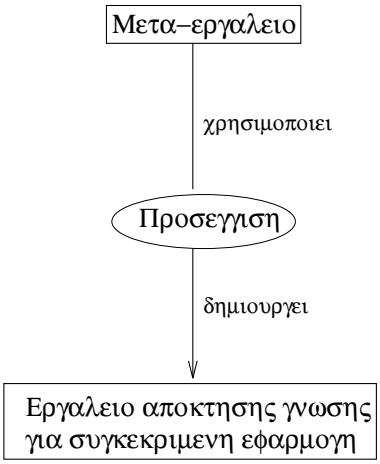
2.3 Βάσεις γνώσης

Αν και ο όρος *γνώση* χρησιμοποιείται ευρέως τα τελευταία χρόνια, έχει καθιερωθεί κυρίως στα προγραμματιστικά συστήματα που υποστηρίζουν κάποιο είδος δηλωτικής γλώσσας [Ull89]. Οι δηλωτικές γλώσσες με τη σειρά τους υποστηρίζουν κάποιο είδος λογικής. Ετσι λοιπόν, κάνοντας λόγο για σύστημα διαχείρισης βάσεων γνώσης (ΣΔΒΓ), εννοούμε ένα σύστημα που υποστηρίζει μια δηλωτική γλώσσα στο ρόλο της γλώσσας διαχείρισης των δεδομένων.

Στα συστήματα αυτά, το πρόβλημα της απόκτησης γνώσης έχει αντιμετωπιστεί με χρήση εξειδικευμένων εργαλείων. Τα εργαλεία αυτά εξυπηρετούν την εφαρμογή για την οποία κατασκευάστηκαν. Για παράδειγμα, σε ΣΔΒΓ πρώτης γενιάς, όπως το MYCIN, προσαρμόστηκαν εργαλεία απόκτησης γνώσης, όπως κειμενογράφοι, συστήματα γραφικής παρουσίασης των περιεχομένων της βάσης και γενικότερα συστήματα επαφής χρήσης, που διευκολύνουν το έργο του χρήστη [MWKE93]. Ετσι δίνεται στο χρήστη η δυνατότητα να επιβλέπει με κατανοητό τρόπο τη

βάση, να εξηγεί τη συμπεριφορά του συστήματος και να τροποποιεί τη γνώση που έχει εισαχθεί.

Ομως το κόστος κατασκευής εξειδικευμένων εργαλείων για την απόκτηση γνώσης για κάθε εφαρμογή είναι μεγάλο. Εξάλλου προσπάθεια για κατασκευή ενός γενικευμένου εργαλείου δε φέρει το επιθυμητό αποτέλεσμα, αφού αυτό δεν μπορεί να καλύψει τις ανάγκες κάθε εφαρμογής, όσον αφορά στην απόκτηση γνώσης. Για τους παραπάνω λόγους, ενδιαφέρον παρουσιάζει η κατασκευή εξειδικευμένων εργαλείων απόκτησης γνώσης, με χρήση μετα-εργαλείων [EM93], [Mus89], [PEEM], [Eri]. Στο σχήμα 2.1 βλέπουμε πως αναπτύσσεται ένα διαλογικό εργαλείο απόκτησης γνώσης με χρήση μετα-εργαλείων.



Σχήμα 2.1: Καθορισμός εργαλείων απόκτησης γνώσης με μετα-εργαλεία

Το μετα-εργαλείο, για να κατασκευάσει ένα εξειδικευμένο εργαλείο απόκτησης γνώσης, χρησιμοποιεί μια από τις προσεγγίσεις: *μεθοδοκεντρική, αφηρημένης αρχιτεκτονικής, οντολογική*

Σύμφωνα με το [EM93], χρησιμοποιούνται τρεις προσεγγίσεις για τον καθορισμό εργαλείων απόκτησης γνώσης:

1. *μεθοδοκεντρική (method oriented)*: επιτρέπει στον αναλυτή να πραγματοποιήσει ένα -δεδομένο- μοντέλο επίλυσης προβλημάτων, αντιστοιχίζοντας έννοιες και σχέσεις του μοντέλου με κατάλληλους όρους του πεδίου εφαρμογής. Ουσιαστικά κατασκευάζεται μια γενική φόρμα εισαγωγής πληροφορίας, που φιλοδοξεί να καλύψει ένα σύνολο εφαρμογών. Μειονεκτεί στο ότι περιορίζεται σε εφαρμογές για τις οποίες είναι κατάλληλο το συγκεκριμένο μοντέλο.

2. *αφηρημένη αρχιτεκτονική (abstract architecture)*, που επιτρέπει στον αναλυτή να καθορίσει τμήματα της αρχιτεκτονικής του εργαλείου απόκτησης γνώσης, ανεξάρτητα από το μοντέλο μεθόδου. Μειονεκτεί στο ότι το εργαλείο απόκτησης γνώσης κατασκευάζεται μόνο με τμήματα που ήδη υπάρχουν (κειμενογράφοι, φόρμες).

3. *οντολογική (ontological)*, που δίνει έμφαση στις έννοιες και τις σχέσεις του πεδίου εφαρμογής. Η ισχύς της προσέγγισης πηγάζει από την οντολογία της εφαρμογής, η οποία καθορίζει τη

λειτουργικότητα του εργαλείου απόκτησης γνώσης. Μειονεκτεί στο ότι πρέπει συνεχώς να γίνονται ρυθμίσεις στο εργαλείο απόκτησης γνώσης, αφού η οντολογία της εφαρμογής, αλλάζει συνεχώς, μεταβάλλοντας τη λειτουργικότητά του.

Στη συνέχεια παρουσιάζουμε τρία χαρακτηριστικά μετα-εργαλεία, που χρησιμοποιούν μια από τις παραπάνω προσεγγίσεις. Είναι τα **Protege**, **Dots** και **Dash** που περιγράφονται στο [EM93].

Το **Protege** είναι ένα διαλογικό εργαλείο, που βοηθά στην πραγματοποίηση μοντέλου επίλυσης προβλημάτων, ενώ αυτομάτως κατασκευάζει εύκολα τροποποιήσιμα εργαλεία απόκτησης γνώσης, βασισμένα σε αυτό το μοντέλο [Mus89]. Με το μοντέλο επίλυσης προβλημάτων αναπαριστάται η μορφή του προβλήματος που το σύστημα καλείται να αντιμετωπίσει. Χρησιμοποιεί τη μεθοδοκεντρική προσέγγιση για την κατασκευή εργαλείων απόκτησης γνώσης. Το **Protege** προήλθε από τη γενίκευση του **Opal**, ενός εξειδικευμένου στη χρήση εργαλείου απόκτησης γνώσης. Το τελευταίο χρησιμοποιήθηκε σε ιατρικές εφαρμογές και συγκεκριμένα για την εξαγωγή συμπερασμάτων σε περιστατικά ασθενειών, και για τη διαχείριση θεραπειών.

Το **Dots** παράγει το εργαλείο απόκτησης γνώσης χρησιμοποιώντας την προσέγγιση της αφηρημένης αρχιτεκτονικής. Διαφέρει από το **Protege** στο ότι εστιάζει στην αρχιτεκτονική του εργαλείου, παρά στο μοντέλο επίλυσης προβλημάτων.

Το **Dash** υποστηρίζει την οντολογική προσέγγιση, ενώ χρησιμοποιείται σαν υποσύστημα του **ProtegeII**. Το τελευταίο παρέχει βιβλιοθήκη από αναχρησιμοποιούμενες μεθόδους επίλυσης προβλημάτων. Η οντολογία που καθορίζει τη λειτουργικότητα του εργαλείου απόκτησης γνώσης που θα παραχθεί, δίνεται σαν είσοδος στο **Dash**. Αυτή αποτελείται από ορισμούς κατάλληλους για δημιουργία εξειδικευμένων εργαλείων. Ξεχωρίζει τη δημιουργία του εργαλείου σε δύο στάδια: Στο πρώτο στάδιο σχεδιάζεται η δομή του διαλόγου και καθορίζεται η αλληλεπίδραση των τμημάτων του. Στο δεύτερο στάδιο καθορίζεται η επαφή χρήσης που θα υλοποιήσει τις δομές του διαλόγου (κειμενογράφοι, μενού επιλογών, φόρμες κα.)

2.4 Οντοκεντρικές βάσεις

Οι οντοκεντρικές βάσεις δεδομένων (ΟΒΔ) είναι μια νέα γενιά βάσεων δεδομένων οι οποίες πετυχαίνουν να περιγράψουν ευκολότερα και με μεγαλύτερη εκφραστικότητα τον πραγματικό κόσμο. Ο χώρος των οντοκεντρικών βάσεων είναι νέος, και για αυτό το λόγο δεν υπάρχει ακόμα τυποποίηση. Το μοντέλο δεδομένων, η λογική οργάνωση των οντοτήτων του πραγματικού κόσμου, των περιορισμών που επιβάλλονται σε αυτές και των μεταξύ τους σχέσεων, δεν έχει σταθεροποιηθεί. Σε γενικές γραμμές όμως βασίζονται στο οντοκεντρικό μοντέλο δεδομένων [PM88] το οποίο παριστάνει τη γνώση μέσω της έννοιας της **οντότητας**, και των μηχανισμών **ταξινόμησης, εξειδίκευσης** και **απόδοσης γνωρίσματος**.

Συχνά οι οντοκεντρικές βάσεις, κυρίως οι εμπορικότερες από αυτές, σχεδιάζονται σαν

επεκτάσεις οντοκεντρικών γλωσσών προγραμματισμού, με σκοπό να παρέχουν δυνατότητα μόνιμης αποθήκευσης, ερωτήσεων, ελέγχου συνδρομικότητας και άλλων υπηρεσιών στις οντότητες των γλωσσών που υποστηρίζουν. Η πολυπλοκότητα του σχήματος των οντοκεντρικών βάσεων, κάνει δύσκολο το θέμα της λογικής τους σχεδίασης. Η ανάγκη για φιλική και αποτελεσματική βοήθεια στη σχεδίασή τους είναι πιο έντονη από την αντίστοιχη για τις σχεσιακές. Γενικότερα ο σχεδιασμός υψηλού επιπέδου επαφών χρήσεως παραμένει μια περιοχή ανοικτή για έρευνα, όχι μόνο για τις οντοκεντρικές βάσεις δεδομένων αλλά και για τις συμβατικές [Kim90].

Ακολουθώντας, στο βαθμό που επιτρέπεται, τη μέθοδο διαλογικής ενημέρωσης των σχεσιακών βάσεων δεδομένων, οι οντοκεντρικές βάσεις χρησιμοποιούν γραφικές μεθόδους, κυρίως για τη διαλογική ενημέρωση του σχήματός τους. Προσφέρονται επίσης τροποποιήσιμα δελτία ενημέρωσης οντοτήτων, που βοηθούν στην ενημέρωση της βάσης. Στη συνέχεια μελετούμε ενδεικτικά κάποια από τα γνωστότερα ΣΔΒΔ που στηρίζονται στο οντοκεντρικό μοντέλο δεδομένων.

2.4.1 GemStone

Το GemStone [BOS91], [ZCC95] είναι ένα από τα πρώτα ΣΔΒΔ που χρησιμοποίησε την έννοια της οντότητας σαν δομικό στοιχείο για την κατασκευή της βάσης του. Αναπτύχθηκε για την ικανοποίηση εμπορικών εφαρμογών αλλά και εφαρμογών κατάλληλων για μηχανικούς λογισμικού. Υποστηρίζει ένα σύνολο γλωσσών για την ενημέρωση της βάσης που διαχειρίζεται. Ετσι προσφέρει δυνατότητα για χρήση γλωσσών προγραμματισμού, όπως η C, C++, Smalltalk, ενώ υποστηρίζει και μια διάλεκτο της Smalltalk, τη GemStone Data Manipulation Language. Τέλος υποστηρίζει ένα σύνολο από εμπορικές σχεσιακές βάσεις (Sybase, Ingres, Oracle, Informix), μεταφράζοντας αποτελέσματα SQL ερωτήσεων που γίνονται σε αυτές, σε αντικείμενα (objects) της οντοκεντρικής βάσης.

Το σύστημα προσφέρει ένα περιβάλλον ανάπτυξης εφαρμογών, το GEODE (GemStone Object Development Environment). Τα εργαλεία που παρέχονται για διαλογική ενημέρωση της βάσης είναι:

- Visual Schema Designer: επιτρέπει την ενημέρωση του σχήματος, κάνοντας τη διαδικασία αυτή εύκολα εποπτεύσιμη. Ο χρήστης μπορεί να δημιουργήσει κλάσεις και σχέσεις μεταξύ τους, είτε σε μορφή κειμένου συμπληρώνοντας κατάλληλη φόρμα, είτε γραφικά σχεδιάζοντας ακμές που αναπαριστούν σχέσεις ανάμεσα σε κόμβους γράφου που αναπαριστούν κλάσεις.

- Visual Forms Designer: διαβάζει τις προδιαγραφές κλάσεων μαζί με ένα σχετικό λεξικό δεδομένων, για να παράγει αυτόματα φόρμες κατάλληλες για απλή εισαγωγή δεδομένων. Οι φόρμες μπορεί να ρυθμιστούν, ως προς την εμφάνισή τους εύκολα, χρησιμοποιώντας ένα πλήθος συστατικών επαφής χρήσης που το εργαλείο παρέχει.

- Visual Program Designer: επιτρέπει στους κατασκευαστές εφαρμογών να δημιουργούν ή να τροποποιούν τη συμπεριφορά μιας εφαρμογής, χωρίς να χρειάζεται να γράψουν κώδικα. Προγράμματα συνθέτονται από τμήματα προγραμμάτων με τρόπο γραφικό.

2.4.2 O_2

Το O_2 [Deux91], [Deux90], [Ban92], [Sol92], [ZCC95] είναι ένα από τα πιο γνωστά και διαδεδομένα συστήματα διαχείρισης οντοκεντρικών βάσεων δεδομένων. Χαρακτηριστικά του είναι ότι υποστηρίζει την έννοια της σύνθετης οντότητας, ενώ υπάρχει σαφής διαχωρισμός του σχήματος από τα δεδομένα. Το σύστημα μπορεί να υποστηρίξει δύο είδη επαφής χρήσης, για την δημιουργία βάσεων: Χρησιμοποιώντας κάποια γλώσσα προγραμματισμού (C ή C++) είτε χρησιμοποιώντας το περιβάλλον O_2.

Το περιβάλλον O_2 προσφέρει ανάμεσα στα άλλα το O_2Look, μια γλώσσα παραγωγής επαφής χρήσεως, και το O_2C, μια γλώσσα για προγραμματισμό και διαχείριση της βάσης. Συνδυάζοντας το O_2Look και O_2C έχει κανείς τη δυνατότητα κατασκευής εφαρμογών. Με το O_2Look υπάρχει η δυνατότητα εμφάνισης καρτών παρουσίασης κάθε οντότητας της βάσης. Μπορούν να κατασκευαστούν ‘‘έτοιμες’’ κάρτες παρουσίασης, ανεξάρτητα από την πολυπλοκότητα της οντότητας (απλή/σύνθετη). Οι κάρτες αυτές δίνουν τη δυνατότητα να τροποποιήσουμε τα κομμάτια της βάσης που παρουσιάζονται. Μπορούν επίσης να οριστούν ‘‘μάσκες’’ για μερική εμφάνιση της οντότητας. Με αυτό τον τρόπο δίνεται η δυνατότητα για μερική τροποποίησή της.

Υπάρχουν τρεις καταστάσεις για την παρουσίαση της οντότητας στην κάρτα: icon, level_one, all_you_can_show. Στην πρώτη κατάσταση, η κάρτα της οντότητας εμφανίζεται με τη μορφή εικονιδίου. Στη δεύτερη, εμφανίζεται η οντότητα και όλες οι οντότητες που αυτή περιέχει, σε επίπεδο βάθους ένα. Στην τρίτη κατάσταση, η κάρτα αναλύει πλήρως τη -σύνθετη- οντότητα, παρουσιάζοντας όλες τις υπο-οντότητές της, σε όλα τα επίπεδα βάθους.

Το O_2Tools, κατασκευασμένο με τα O_2Look και O_2C, είναι ένα γραφικό προγραμματιστικό περιβάλλον, που υποστηρίζει την ανάπτυξη εφαρμογών. Επιτρέπει στο χρήστη να διερευνά και να ενημερώνει τα δεδομένα αλλά και το σχήμα της βάσης. Οσον αφορά στην τροποποίηση του σχήματος ειδικότερα, το O_2Tools μπορεί να λειτουργήσει σε τρεις καταστάσεις:

1. αυτόματη κατάσταση, όπου τα μη συνεπή προς το σχήμα δεδομένα επαναμεταφράζονται αυτόματα.

2. σταθερή κατάσταση, όπου τα μη συνεπή προς το σχήμα δεδομένα χαρακτηρίζονται άκυρα και δε χρησιμοποιούνται πλέον.

3. ελεύθερη κατάσταση, όπου η πλήρης ευθύνη για συνέπεια σχήματος και δεδομένων αφήνεται στον προγραμματιστή.

2.4.3 ObjectStore

Το ΣΔΒΔ ObjectStore [Sol92], [ZCC95] χρησιμοποιεί C ή C++ για την ανάπτυξη της βάσης που διαχειρίζεται. Εναλλακτικά μπορεί να δημιουργήσει κανείς το σχήμα της βάσης με το Schema

Designer, ένα διαλογικό γραφικό εργαλείο για ανάπτυξη και παρουσίαση κλάσεων σε επίπεδο σχήματος.

2.4.4 Ontos

Το ΣΔΒΔ Ontos [Sol92], [ZCC95] προσφέρει τις γλώσσες C++ και Ontos SQL για την ανάπτυξη της βάσης που διαχειρίζεται. Με το DBDesigner, ένα εργαλείο σχεδίασης του σχήματος της βάσης, αλλά και εξερεύνησής της, δίνει τη δυνατότητα για διαλογική ανάπτυξη της εφαρμογής.

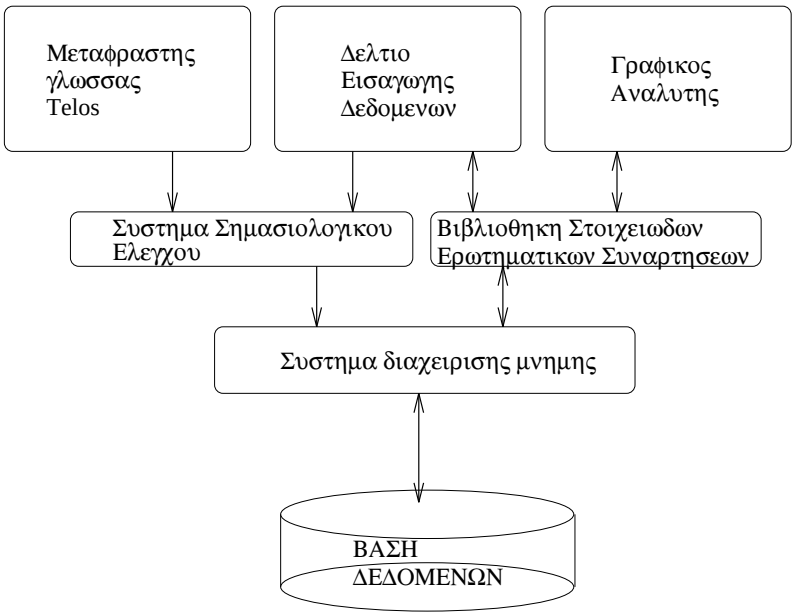
2.5 Το Σύστημα Σημασιολογικού Ευρετηριασμού

Το Σύστημα Σημασιολογικού Ευρετηριασμού (ΣΣΕ) είναι ένα εργαλείο περιγραφής και τεκμηρίωσης στενά συνδεδεμένων μεταξύ τους δεδομένων, εννοιών και πολύπλοκων σχέσεων [CDb]. Η περιγραφή αυτή γίνεται στο μοντέλο δεδομένων της γλώσσας παράστασης γνώσης Telos, με τη βοήθεια του μεταφραστή της γλώσσας [DKT93] ή του Δελτίου Εισαγωγής Δεδομένων [DT94]. Και τα δυο αυτά εργαλεία επικοινωνούν με το σύστημα σημασιολογικού ελέγχου [DDPM93], το οποίο δίνει τη δυνατότητα ενημερώσεων. Για την άντληση της πληροφορίας το ΣΣΕ παρέχει μια Βιβλιοθήκη Στοιχειωδών Ερωτηματικών Συναρτήσεων (ΒΣΕΣ) [Ντα93b], [DD94]. Τόσο αυτή η βιβλιοθήκη, όσο και το σύστημα σημασιολογικού ελέγχου, επικοινωνούν με το σύστημα διαχείρισης μνήμης του ΣΣΕ [Γεω94b], και αυτό με τη βάση. Το περιβάλλον συμπληρώνεται με ένα σύστημα επαφής χρήσης που ονομάζουμε γραφικό αναλυτή [BDD+92]. Το εργαλείο αυτό χρησιμεύει στην παρουσίαση της πληροφορίας τόσο σε γραφική μορφή όσο και σε μορφή κειμένου. Το περιβάλλον του Σημασιολογικού Συστήματος Ευρετηριασμού παρουσιάζεται στο σχήμα 2.2.

2.5.1 Η γλώσσα Telos

Η Telos είναι μια γλώσσα παράστασης γνώσης, που περιγράφεται αναλυτικά στο [MBJK90]. Στην παρούσα εργασία θα αναφερόμαστε στην έκδοση του δομικού μέρους της γλώσσας, όπως αναπτύχθηκε από την Ομάδα Πληροφοριακών Συστημάτων και Τεχνολογίας Λογισμικού του ΙΠ-ΙΤΕ και αναλυτικά περιγράφεται στο [DKT93].

Οι μηχανισμοί που χρησιμοποιεί η γλώσσα είναι ανάλογοι με αυτούς των σημασιολογικών δικτύων. Οι στοιχειώδεις δομές της είναι οντότητες και σχέσεις μεταξύ των οντοτήτων που μπορούν να συνδυαστούν για να παραστήσουν πολυπλοκότερες δομές. Οι οντότητες που περιγράφονται στην Telos είναι είτε ατομικές οντότητες είτε κλάσεις. Οι βασικοί μηχανισμοί της γλώσσας είναι οι σχέσεις **ταξινόμησης**, **εξειδίκευσης** και **απόδοσης γνωρίσματος**. Οι ιεραρχίες ταξινόμησης και εξειδίκευσης μπορούν να είναι απεριόριστες και πολλαπλές.



Σχήμα 2.2: Το περιβάλλον του Συστήματος Σημασιολογικού Ευρετηριασμού

2.5.1.1 Η δομή της γλώσσας

Το σημασιολογικό δίκτυο που φτιάχνουμε με το μοντέλο δεδομένων της γλώσσας Telos αποτελείται από οντότητες και σχέσεις μεταξύ τους. Κάποιες από αυτές είναι ανεξάρτητες οντότητες αφού μπορεί να υπάρχουν στη βάση χωρίς να χρειάζονται ‘‘υποστήριξη’’ από άλλες. Αυτές τις καλούμε *κόμβους*. Κάποιες άλλες είναι εξαρτημένες οντότητες γιατί εκκινούν από μια οντότητα, χωρίς την οποία δεν μπορούν να ‘‘σταθούν’’ στο σημασιολογικό δίκτυο. Αυτές τις καλούμε *γνωρίσματα*. Υπάρχουν σχέσεις μεταξύ οντοτήτων, οι οποίες δεν αποτελούν οι ίδιες οντότητες[1]. Πρόκειται για τους *συνδέσμους περίπτωσης* και *εξειδίκευσης*.

Η *σχέση περίπτωσης* είναι σχέση του μέρους προς το όλον. Μια οντότητα Α λέμε ότι είναι *περίπτωση* της οντότητας Β, αν υπάρχει ένας σύνδεσμος περίπτωσης από την Α στη Β. Επίσης η Β λέγεται και *κλάση* της Α. Ειδικά για τα γνωρίσματα, ο όρος κλάση μπορεί να αντικατασταθεί με τον όρο *κατηγορία*. Ετσι, κατηγορία ενός γνωρίσματος είναι η κλάση στην οποία ανήκει. Η κατηγορία είναι η ίδια ένα γνώρισμα.

Μια οντότητα μπορεί να ανήκει σε περισσότερες από μια κλάσεις (πολλαπλή ταξινόμηση). Ακόμα, χρησιμοποιεί τα γνωρίσματα των κλάσεών της για κατηγορίες των δικών της γνωρισμάτων. Πολλές φορές, χρησιμοποιούμε φράσεις όπως ‘‘οι κατηγορίες γνωρισμάτων της οντότητας Ο’’ ή ‘‘η οντότητα Ο με τις κατηγορίες γνωρισμάτων ...’’ εννοώντας τα γνωρίσματα όλων των κλάσεων της εν λόγω οντότητας. Ετσι μια οντότητα, κόμβος είτε γνώρισμα, μπορεί να **έχει** κατηγορίες γνωρισμάτων, τα γνωρίσματα όλων των κλάσεών της, ενώ ένα γνώρισμα μπορεί να **ανήκει** σε κατηγορίες γνωρισμάτων, αυτές των οποίων είναι περίπτωση.

[1]Μπορούμε να θεωρήσουμε τα γνωρίσματα προνομιακές σχέσεις αφού έχουν υλοποιηθεί σαν οντότητες.

Η *σχέση εξειδίκευσης* είναι σχέση συνόλου προς το υποσύνολό του. Μια οντότητα Α λέμε ότι είναι *εξειδίκευση* της οντότητας Β, αν υπάρχει ένας σύνδεσμος εξειδίκευσης από την Α στη Β. Εναλλακτικά λέμε ότι η οντότητα Β είναι *γενίκευση* της Α. Η Β λέγεται *υπερκλάση* της Α, ενώ η Α λέγεται *υποκλάση* της Β. Μια οντότητα μπορεί να ανήκει σε περισσότερες από μία υπερκλάσεις (πολλαπλή εξειδίκευση). Μια οντότητα κληρονομεί **όλα** τα γνωρίσματα των υπερκλάσεών της (αυστηρή κληρονόμηση).

Οι κλάσεις στις οποίες έχει ταξινομηθεί ένας κόμβος λέγονται *άμεσες* κλάσεις. Χρησιμοποιούμε αυτό τον όρο για να τις ξεχωρίζουμε από τις *έμμεσες* κλάσεις του. Αυτές τις συμπεραίνει το σύστημα και είναι όλες οι υπερκλάσεις των κλάσεων της οντότητας. Παρόμοια ξεχωρίζουμε τις άμεσες από τις έμμεσες υπερκλάσεις της οντότητας, ενώ ανάλογα διακρίνουμε τις περιπτώσεις και τις υποκλάσεις της. Αν δεν κάνουμε διάκριση ανάμεσα σε άμεσες και έμμεσες, ενοούμε γενικά την ένωσή τους.

Κάθε οντότητα χαρακτηρίζεται, κατά τη δημιουργία της, ως **κόμβος (Individual)** ή **γνώρισμα (Attribute)**, ενώ αυτός ο ‘‘χαρακτηρισμός’’ δεν μπορεί να αλλάξει σε όλη τη διάρκεια ‘‘ζωής’’ της στο σύστημα. Μια οντότητα συνδέεται υποχρεωτικά με ένα μοναδικό -εσωτερικό- αναγνωριστικό. Υποχρεωτικό είναι για τους κόμβους το λογικό (εξωτερικό) τους όνομα, ενώ για τα γνωρίσματα είναι προαιρετικό. Ενα αναλλοίωτο χαρακτηριστικό μιας οντότητας, το οποίο δίνεται κατά τη δημιουργία της, είναι το επίπεδο ταξινόμησης. Εχουμε στη διάθεσή μας θεωρητικά όσα επίπεδα χρειαστούμε, με τις ατομικές οντότητες να βρίσκονται στο επίπεδο **Token**, τις κλάσεις στο επίπεδο **S_Class**, τις μετα-κλάσεις στο επίπεδο **M1_Class** κοκ.

Για τα γνωρίσματα ειδικότερα, το επίπεδο ταξινόμησής τους μπορεί να είναι ίσο ή μικρότερο του ελαχίστου των επιπέδων ταξινόμησης της οντότητας εκκίνησης και της τιμής γνωρίσματός τους. Αν δεν καθοριστεί διαφορετικά, θεωρούμε ότι είναι ίσο με το ελάχιστο των επιπέδων ταξινόμησης.

Τα γνωρίσματα μπορούν να εκκινούν από οντότητες (κόμβους ή γνωρίσματα) αλλά μπορεί να καταλήγουν μόνο σε κόμβους ή πρωτογενείς τύπους. Οι πρωτογενείς τύποι είναι οι ακέραιοι (**Telos_Integer**), οι πραγματικοί (**Telos_Real**) και οι ορμαθοί χαρακτήρων (**Telos_string**). Οι περιπτώσεις τους δεν μπορούν να δημιουργηθούν, αλλά θεωρούμε ότι προϋπάρχουν.

Ενα σύνολο από κλάσεις αποτελούν τον αρχικό πληθυσμό κάθε βάσης. Πάνω σε αυτές ‘‘χτίζουμε’’ την εφαρμογή μας. Αυτές είναι οι *κλάσεις συστήματος*[2], που αποτελούνται από τα επίπεδα ταξινόμησης (**Token**, **S_Class**, **M1_Class**, κ.ο.κ), τους πρωτογενείς τύπους (**Telos_string**, **Telos_Integer**, **Telos_Real**) και τις ‘‘γενικές’’ κλάσεις οντοτήτων (**Individual** και **Attribute**). Στη συνέχεια, όταν αναφερόμαστε σε κλάσεις, θα εννοούμε *κλάσεις ορισμένες από χρήστη*, εκτός αν ρητώς αναφέρεται διαφορετικά.

[2] ονομάζονται έτσι για να διακρίνονται από τις *κλάσεις ορισμένες από χρήστη*

2.5.1.2 Δομικοί περιορισμοί

Στις βάσεις δεδομένων που υλοποιούνται με το μοντέλο δεδομένων της Telos, και σύμφωνα με το [DKT93], έχουν εφαρμοστεί ένα σύνολο δομικών περιορισμών. Το σύστημα σημασιολογικού ελέγχου του ΣΣΕ φροντίζει για την τήρησή τους. Οι περιορισμοί αυτοί είναι :

1. Το λογικό όνομα των κόμβων είναι υποχρεωτικό και μοναδικό στη βάση.

2. Το λογικό όνομα των γνωρισμάτων είναι προαιρετικό, αλλά μοναδικό στην εμβέλεια της οντότητας εκκίνησής τους. Δηλαδή, δυο γνωρίσματα που εκκινούν από την ίδια οντότητα δεν μπορούν να έχουν το ίδιο λογικό όνομα. Αν ένα γνώρισμα δεν έχει όνομα, τότε πρέπει να έχει τιμή γνωρίσματος ή να ανήκει σε κατηγορία, διαφορετική από κάθε άλλο γνώρισμα χωρίς όνομα που εκκινεί από την ίδια οντότητα.

3. Μια οντότητα μπορεί να γίνει περίπτωση μιας άλλης, μόνο αν και οι δύο είναι του ίδιου τύπου και η πρώτη βρίσκεται ένα επίπεδο ταξινόμησης χαμηλότερα από τη δεύτερη.

4. Μια οντότητα μπορεί να γίνει εξειδίκευση μιας άλλης, μόνο αν και οι δύο είναι του ίδιου τύπου[3]. Πρέπει επίσης αυτές οι δυο οντότητες να βρίσκονται στο ίδιο επίπεδο ταξινόμησης. Σχέσεις εξειδίκευσης δεν μπορούμε να έχουμε σε ατομικό επίπεδο. Τέλος δεν επιτρέπεται να δημιουργηθεί κύκλος από συνδέσμους εξειδίκευσης.

5. Διαγραφή μιας οντότητας επιτρέπεται μόνο όταν αυτή δεν έχει γνωρίσματα που εκκινούν ή καταλήγουν σε αυτήν και δεν έχει άμεσες περιπτώσεις.

6. Ειδικότερα για τα γνωρίσματα ισχύει επιπλέον ένας περιορισμός: Για να μπορέσουμε να ταξινομήσουμε ένα γνώρισμα σε μία κατηγορία *C*, πρέπει η οντότητα εκκίνησής του να είναι περίπτωση της οντότητας εκκίνησης της κατηγορίας *C*. Επίσης πρέπει η τιμή του γνωρίσματος να είναι περίπτωση της τιμής γνωρίσματος της κατηγορίας *C*.
   Ανάλογοι κανόνες ισχύουν όταν θέλουμε να γενικεύσουμε κατηγορία γνωρισμάτων κάνοντάς την υποκλάση μιας άλλης.

2.5.2 Στοιχειώδεις ενημερώσεις

Από το Σύστημα Σημασιολογικού Ευρετηριασμού μας ενδιαφέρει ιδιαίτερα η πλευρά της ενημέρωσης πληροφορίας. Σε αυτήν την ενότητα περιγράφουμε τις στοιχειώδεις ενημερώσεις, τις πιο απλές δηλαδή ενημερώσεις που μπορεί κάποιος να εφαρμόσει σε μια οντότητα της βάσης. Σχετική μελέτη έχει γίνει στο [Τζί95b].

Μερικές από τις ενημερώσεις αυτές αναφέρονται γενικά σε οντότητες, άλλες αποκλειστικά σε κόμβους και άλλες αποκλειστικά σε γνωρίσματα. Παραθέτουμε τις στοιχειώδεις ενημερώσεις μαζί με

[3] και οι δύο είναι κόμβοι ή και οι δύο είναι γνωρίσματα

τα ορίσματά τους και μια σύντομη περιγραφή τους. Σημειώνεται ότι όπου αναφέρεται στα ορίσματα των πράξεων *οντότητα εκκίνησης* ή απλώς *οντότητα* σημαίνει πως έχουμε κόμβο ή γνώρισμα, ενώ όπου αναφέρεται *τιμή γνωρίσματος* σημαίνει πως έχουμε κόμβο ή πρωτογενή τύπο.

- Δημιουργία κόμβου (*λογικό όνομα*, *επίπεδο ταξινόμησης*)
  Δημιουργία κόμβου με όνομα το *λογικό όνομα* και επίπεδο ταξινόμησης το *επίπεδο ταξινόμησης*.

- Διαγραφή κόμβου (*λογικό όνομα*)
  Διαγραφή του κόμβου με όνομα το *λογικό όνομα*.

- Δημιουργία γνωρίσματος (*λογικό όνομα*, *επίπεδο ταξινόμησης*, *οντότητα εκκίνησης*, *τιμή γνωρίσματος*)
  Δημιουργία γνωρίσματος με όνομα το *λογικό όνομα*, οντότητα εκκίνησης την *οντότητα εκκίνησης* και τιμή γνωρίσματος την *τιμή γνωρίσματος*. Η παράμετρος *λογικό όνομα* είναι προαιρετική. Αν δεν δοθεί λογικό όνομα για το γνώρισμα, τότε το σύστημα αναλαμβάνει να δώσει ένα αυτόματα παραγόμενο όνομα - περίπτωση γνωρίσματος χωρίς όνομα. Το *επίπεδο ταξινόμησης* είναι κι αυτό προαιρετικό, μόνο όμως στην περίπτωση που αυτό μπορεί να εξαχθεί από την *οντότητα εκκίνησης* και/ή την *τιμή γνωρίσματος*. Αν όχι, τότε υποχρεωτικά πρέπει να καθοριστεί.

- Διαγραφή γνωρίσματος(*λογικό όνομα*, *οντότητα εκκίνησης*, *τιμή γνωρίσματος*)
  Διαγραφή γνωρίσματος με όνομα το *λογικό όνομα* (αν υπάρχει τέτοιο), που έχει οντότητα εκκίνησης την *οντότητα εκκίνησης* και τιμή γνωρίσματος την *τιμή γνωρίσματος*.

- Μετονομασία οντότητας (*οντότητα*, *λογικό όνομα*)
  Μετονομασία του λογικού ονόματος της οντότητας (η οποία καθορίζεται από το *οντότητα*) με το *λογικό όνομα*.

- Δημιουργία συνδέσμου περίπτωσης (*οντότηταA*, *οντότηταB*)
  Δημιουργία συνδέσμου περίπτωσης από την *οντότηταA* προς την *οντότηταB*. Ετσι η *οντότηταB* προσθέτει στις περιπτώσεις της την *οντότηταA*.

- Διαγραφή συνδέσμου περίπτωσης (*οντότηταA*, *οντότηταB*)
  Διαγραφή συνδέσμου περίπτωσης από την *οντότηταA* προς την *οντότηταB*. Ετσι η *οντότηταB* διαγράφει από τις περιπτώσεις της την *οντότηταA*.

- Δημιουργία συνδέσμου εξειδίκευσης (*οντότηταA*, *οντότηταB*)
  Δημιουργία συνδέσμου εξειδίκευσης από την *οντότηταA* προς την *οντότηταB*. Ετσι η *οντότηταB* προσθέτει στις υποκλάσεις της την *οντότηταA*.

- Διαγραφή συνδέσμου εξειδίκευσης (*οντότηταA*, *οντότηταB*)
  Διαγραφή συνδέσμου εξειδίκευσης από την *οντότηταA* προς την *οντότηταB*. Ετσι η *οντότηταB* διαγράφει από τις υποκλάσεις της την *οντότηταA*.

2.5.3 Μαζική εισαγωγή δεδομένων

Στο Σύστημα Σημασιολογικού Ευρετηριασμού η μαζική εισαγωγή δεδομένων γίνεται με τη βοήθεια του μεταφραστή της γλώσσας Telos. Εχει οριστεί ένα συντακτικό από προτάσεις [DDPM93], [DKT93], που υλοποιούν το μοντέλο δεδομένων της γλώσσας. Οι προτάσεις αυτές, τοποθετημένες σε νοηματικά πλήρεις δοσοληψίες, ενημερώνουν τη βάση. Ο έλεγχος της σημασιολογικής ορθότητας των δοσοληψιών γίνεται από το σύστημα σημασιολογικού ελέγχου, το οποίο με τη σειρά του ενημερώνει το υποσύστημα διαχείρισης μνήμης του ΣΣΕ.

2.5.4 Διαλογική εισαγωγή δεδομένων

Στο Σύστημα Σημασιολογικού Ευρετηριασμού η διαλογική εισαγωγή δεδομένων γίνεται με τη βοήθεια του ΔΕΔ. Η λειτουργικότητά του εργαλείου παρουσιάζεται στα επόμενα κεφάλαια.

Η ιδέα της κατασκευής ενός διαλογικού εργαλείου για την εισαγωγή δεδομένων δεν είναι καινούργια. Στα [Πατ93a], [CP], [CDP+92] βλέπουμε μια πρώτη προσέγγιση για την κατασκευή ενός τέτοιου εργαλείου, που εξυπηρετούσε την εισαγωγή δεδομένων, σαν τμήμα του Εργαλείου Επιλογής (ΕΕ). Το τελευταίο χρησιμοποιήθηκε στην επιλογή τμημάτων λογισμικού και αποτέλεσε τον πρόγονο του ΣΣΕ. Οι δυνατότητες που είχε το διαλογικό εργαλείο εισαγωγής δεδομένων του ΕΕ ήταν περιορισμένες. Για έναν υπάρχοντα κόμβο, υπήρχε η δυνατότητα να εισαχθεί πληροφορία που αφορά ταξινόμησή του σε κλάσεις και δημιουργία γνωρισμάτων που εκκινούν από αυτόν. Εναλλακτικά, μπορούσε ο κόμβος να δημιουργείται για πρώτη φορά στη βάση.

Αυτό το εργαλείο είχε ελλείψεις στη λειτουργικότητά του, αφού δεν υποστήριζε πλήρες σύνολο από πράξεις ενημέρωσης. Επίσης η επαφή χρήσης του ήταν αρκετά εξειδικευμένη για την εφαρμογή για την οποία δοκιμάστηκε, τη Βάση Περιγραφής Λογισμικού[4].

Τόσο το ΔΕΔ, όσο και το εργαλείο εισαγωγής δεδομένων του ΕΕ, έχουν κοινό περιβάλλον ανάπτυξης (σχήμα 2.2). Μια διαφορά είναι ότι το ΔΕΔ είναι ανεξάρτητο από το γραφικό αναλυτή της βάσης, για να μπορεί να καλείται χωρίς την υποστήριξή του. Επίσης έχει ένα μοντέλο διεργασιών που το καθοδηγεί, ένα πλήρες σύνολο από πράξεις ενημέρωσης, ενώ η επαφή χρήσης που χρησιμοποιεί είναι τελείως διαφορετική, σχεδιασμένη από την αρχή, για να μπορεί να ικανοποιεί τις απαιτήσεις πολλών εφαρμογών. Για όλα αυτά θα μιλήσουμε στα επόμενα κεφάλαια.

[4]Είναι μια βάση υλοποιημένη στο μοντέλο δεδομένων της Telos, και περιγράφει τμήματα λογισμικού με σκοπό την αναχρησιμοποίησή τους.

2.6 Προσεγγίσεις στη διαλογική ενημέρωση

Για την κατασκευή συστημάτων επαφής χρήσης μπορούν να χρησιμοποιηθούν δύο γενικού τύπου προσεγγίσεις, η οντοκεντρική και η διεργασιοκεντρική. Η πρώτη δίνει τη δυνατότητα στο χρήστη να θέσει σαν κέντρο του ενδιαφέροντός του μια οντότητα, ένα στοιχείο της επαφής χρήσης και του επιτρέπει να το χειριστεί όπως εκείνος επιθυμεί, προκειμένου να πραγματοποιήσει το στόχο του. Η δεύτερη θέτει σαν κέντρο του ενδιαφέροντος του χρήστη μια διεργασία, μια ανωτέρου επιπέδου λειτουργία που αναλύεται σε απλούστερες. Προκειμένου να πραγματοποιηθεί αυτή, το σύστημα μπορεί να περάσει από διάφορα στάδια. Στην τελευταία αυτή περίπτωση ο χρήστης, βοηθούμενος από το σύστημα, καλείται να πραγματοποιήσει σύνολο ενεργειών που θα υλοποιήσουν τη διεργασία.

Η διαλογική ενημέρωση δεν θα μπορούσε να ξεφύγει από το γενικό πλαίσιο, που μόλις αναφέραμε. Ετσι λοιπόν υφίσταται η διάκριση ανάμεσα στην οντοκεντρική και τη διεργασιοκεντρική προσέγγιση της διαλογικής ενημέρωσης βάσεων δεδομένων. Αυτές οι δύο έχουν ορισμένες βασικές διαφορές, που τις καθιστούν χρήσιμες για την επίτευξη διαφορετικών στόχων στις εφαρμογές στις οποίες χρησιμοποιούνται. Στη συνέχεια αναλύουμε τις δύο αυτές προσεγγίσεις σε σχέση με τη διαλογική ενημέρωση. Στο ΔΕΔ χρησιμοποιήσαμε τη διεργασιοκεντρική προσέγγιση και εξηγούμε τους λόγους.

2.6.1 Οντοκεντρική προσέγγιση

Κρίνεται επιθυμητό σε πολλές εφαρμογές, κυρίως στη φάση ανάπτυξης του μοντέλου τους, να ενημερώνουμε την πληροφορία με τρόπο οντοκεντρικό. Θέλουμε να έχουμε σαν κέντρο της διαδικασίας ενημέρωσης μια οντότητα, ένα δομικό στοιχείο της βάσης, και σε αυτήν να μπορούμε να εκτελούμε όλες τις δυνατές πράξεις ενημέρωσης που το μοντέλο δεδομένων επιτρέπει. Αυτός είναι ο συνήθης τρόπος που ένας ειδικός χρήστης-κατασκευαστής μοντέλων, σκέφτεται για να δημιουργήσει ένα μοντέλο εφαρμογής. Η οντοκεντρική προσέγγιση είναι ιδιαίτερα φιλική στους χρήστες που έχουν εξοικειωθεί στην κατασκευή μοντέλων, με τη μορφή σημασιολογικών δικτύων.

Ενα άλλο πλεονέκτημα της προσέγγισης, είναι η δυνατότητά της να χρησιμοποιείται για διπλό σκοπό: Παρουσίαση και ενημέρωση πληροφορίας. Στις οντοκεντρικές βάσεις ειδικότερα, είναι συνήθης η τακτική να παρουσιάζεται πληροφορία σε σχέση με μια οντότητα. Ετσι κάρτες παρουσίασης οντοτήτων μπορούν να μετατραπούν σε κάρτες ενημέρωσης, δίνοντας τη δυνατότητα να τροποποιούμε τα παρουσιαζόμενα ‘‘χαρακτηριστικά’’ τους. Επίσης, με αυτόν τον τρόπο, εκμεταλλευόμαστε τη δυνατότητα διάσχισης του δικτύου των οντοκεντρικών βάσεων, μετακινούμενοι από οντότητα σε οντότητα. Στο [Τζί95b] προτείνεται η κατασκευή ενός τέτοιου συστήματος για το ΣΣΕ.

2.6.2 Διεργασιοκεντρική προσέγγιση

Προσπαθώντας να προσεγγίσουμε τον τρόπο σκέψης του χρήστη που εισάγει δεδομένα διαλογικά σε μια βάση, διαπιστώσαμε πως πολλές φορές αυτό που επιθυμεί είναι η επίτευξη ενός στόχου. Ο στόχος αυτός μπορεί να πραγματοποιηθεί μέσα από μια καλά ορισμένη διεργασία ενημέρωσης. Είναι λοιπόν λογικό να θεωρήσουμε ως κέντρο της προσέγγισης τη διεργασία ενημέρωσης και αυτήν να προσπαθήσουμε να περιγράψουμε. Η διεργασία πρέπει να περιέχει ένα σύνολο οντοτήτων κι ένα σύνολο ενημερώσεων, που ο συνδυασμός τους (ενημέρωση X, σε οντότητα Y) να οδηγεί στην επίτευξη του στόχου που η διεργασία καλείται να πραγματοποιήσει. Βέβαια κι εδώ, όπως και στην προηγούμενη προσέγγιση, μια οντότητα θα ενημερωθεί τελικά. Ομως η ενημέρωση δεν έχει σαν κέντρο αυτήν την οντότητα. Κέντρο της είναι η διεργασία που ο χρήστης επιλέγει, για την επίτευξη του στόχου του. Μέσω της διεργασίας, ο χρήστης μπορεί να ενημερώσει οντότητες ‘‘σχετικές’’ με αυτήν. Επίσης, στη διεργασία μπορούν να οριστούν και ‘‘σχετικές’’ με αυτήν ενημερώσεις.

Ενα πλεονέκτημα της μεθόδου είναι ότι προσφέρει καθοδήγηση στο χρήστη, όσον αφορά στις οντότητες που μπορεί και πρέπει να ενημερώσει για την επίτευξη του στόχου του. Ενα άλλο πλεονέκτημα είναι ότι προσφέρει προστασία δεδομένων, αφού η διεργασία μπορεί να ενημερώσει μόνο ‘‘σχετικές’’ με αυτήν οντότητες. Επίσης, οι ενημερώσεις της διεργασίας μπορεί να περιοριστούν στις απόλυτα επιτρεπτές και ασφαλείς.

Η διεργασιοκεντρική προσέγγιση έχει καλύτερα αποτελέσματα, όταν μπορούμε να απομονώσουμε τμήματα της βάσης που μπορούν να ενημερωθούν ανεξάρτητα από άλλα, με προδιαγεγραμμένες λειτουργίες. Μπορούν τότε να κατασκευαστούν διεργασίες που ενημερώνουν ‘‘σχετικές’’ οντότητες με κατάλληλα εξειδικευμένες λειτουργίες. Γενικές διεργασίες τύπου ‘‘Ενημέρωσε τη βάση’’, αν και μπορούν να χρησιμοποιηθούν, δε βοηθούν ουσιαστικά το χρήστη. Σε αυτές τις περιπτώσεις, πρέπει κανείς να είναι γνώστης των περιεχομένων του μοντέλου εφαρμογής για να εντοπίσει την οντότητα που τελικά θα ενημερώσει, ενώ οι λειτουργίες γίνονται αναγκαστικά γενικές, για να έχουν ευρύ φάσμα δυνατοτήτων. Από την άλλη μεριά βέβαια, αυτές οι γενικές διεργασίες βοηθούν τον ειδικό χρήστη, προσφέροντάς του ελευθερία κινήσεων χωρίς περιορισμούς στις ενημερώσεις.

Οπως είπαμε ήδη, η προσέγγιση που χρησιμοποιείται από το ΔΕΔ είναι η διεργασιοκεντρική. Αυτή η επιλογή έγινε όταν παρουσιάστηκε η ανάγκη ενημέρωσης μιας εφαρμογής του ΣΣΕ, της SIB Class Management System. Οι απαιτήσεις που είχε αυτή η εφαρμογή στη διαλογική της ενημέρωση ταίριαζαν στη διεργασιοκεντρική προσέγγιση, οπότε και υιοθετήθηκε. Με αυτήν δόθηκε λύση στο πρόβλημα της διαλογικής ενημέρωσης για τη συγκεκριμένη εφαρμογή. Διαπιστώθηκε πως η ίδια προσέγγιση μπορεί να χρησιμοποιηθεί για την εξυπηρέτηση κι άλλων εφαρμογών του ΣΣΕ. Ετσι, όπως αναφέρουμε στο κεφάλαιο 7, το ΔΕΔ δοκιμάστηκε και σε άλλες εφαρμογές του ΣΣΕ, στις οποίες άλλοτε λιγότερο και άλλοτε περισσότερο ταίριαζε η προσέγγιση που χρησιμοποιεί.

2.7 Σύγκριση με παρόμοια συστήματα

Τα συστήματα διαχείρισης βάσεων δεδομένων αντιμετωπίζουν το θέμα της διαλογικής ενημέρωσης κυρίως σαν θέμα κατασκευής ενός συστήματος επαφής χρήσης. Ειδικότερα οι οντοκεντρικές βάσεις που παρουσιάστηκαν συνοπτικά στην ενότητα 2.4, αντιμετωπίζουν τη διαλογική ενημέρωση με την κατασκευή δελτίων τροποποίησης οντοτήτων σε επίπεδο δεδομένων (O_2, GemStone), ενώ σε επίπεδο σχήματος συχνά χρησιμοποιούνται γραφικές μέθοδοι (GemStone, ObjectStore, Ontos). Στα συστήματα αυτά υπάρχει σαφής διάκριση του σχήματος της βάσης από τα δεδομένα της, γι' αυτό αντιμετωπίζονται με διαφορετικό τρόπο.

Η προσέγγιση που χρησιμοποιείται για την ενημέρωσή τους μπορεί να χαρακτηριστεί οντοκεντρική, αφού δίνεται η δυνατότητα πλοήγησης της βάσης και για κάθε οντότητα δίνεται στο χρήστη η δυνατότητα να την ενημερώνει με πράξεις που το μοντέλο δεδομένων επιτρέπει. Για την O_2 πιο συγκεκριμένα, κάθε τμήμα της οντότητας που είναι ορατό, είναι και ενημερώσιμο. Ο καθορισμός των εμφανίσιμων (άρα και ενημερώσιμων) τμημάτων της οντότητας γίνεται με χρήση προγραμματιστικών μεθόδων.

Στη δικιά μας θεώρηση το θέμα της διαλογικής ενημέρωσης δεν είναι μόνο θέμα κατασκευής ενός συστήματος επαφής χρήσης. Χρησιμοποιείται ένα μοντέλο διεργασιών που βασίζεται στη διεργασιοκεντρική προσέγγιση (δες κεφάλαιο 3), με τη βοήθεια του οποίου προδιαγράφονται οι ενημερώσεις που μπορούν να επιτευχθούν από τους χρήστες. Η διεργασία είναι αυτή που καθορίζει ποιες οντότητες θα ενημερωθούν, ποια τμήματά της συγκεκριμένα (κλάσεις, υπερκλάσεις ή γνωρίσματα) και με ποιες λειτουργίες. Είναι χαρακτηριστικό ότι δεν χρειάζεται να γίνει χρήση γλωσσών προγραμματισμού για την προσαρμογή του συστήματος στις ανάγκες της εφαρμογής που καλείται να ενημερώσει· αρκεί η κατάλληλη εξειδίκευση του μοντέλου διεργασιών. Ακόμα δίνεται η δυνατότητα για επιβολή περιορισμών ακεραιότητας ανάλογα με τις ανάγκες της εφαρμογής (δες κεφάλαιο 5), κάτι που δεν είναι εφικτό από άλλα συστήματα. Μια ιδιαιτερότητα του μοντέλου δεδομένων της Telos είναι ότι δεν κάνει διαχωρισμό σχήματος και δεδομένων, αφού τα θεωρεί οντότητες διαφορετικού επιπέδου ταξινόμησης. Για το λόγο αυτό το ΔΕΔ τα μεταχειρίζεται ισότιμα, ενημερώνοντάς τα με κοινές κάρτες ενημέρωσης, σε αντίθεση με τη διάκριση που επιβάλλουν άλλα συστήματα.

Κεφάλαιο 3

Μοντέλο Διεργασιών Ενημέρωσης

Για τις ανάγκες της διαλογικής ενημέρωσης του ΣΣΕ αναπτύχθηκε ένα Μοντέλο Διεργασιών Ενημέρωσης (ΜΔΕ)[1] [DT94]. Η παράστασή του έχει γίνει στο μοντέλο δεδομένων της γλώσσας παράστασης γνώσης Telos. Κάποιες επεκτάσεις κυρίως στην ερμηνεία του, καθώς και η χρησιμοποίησή του για την κατασκευή ενός συστήματος διαλογικής ενημέρωσης, αποτελούν μέρος της παρούσας εργασίας.

Βασική έννοια του μοντέλου είναι η διεργασία ενημέρωσης που προδιαγράφει το είδος της εργασίας που ο χρήστης μπορεί να πραγματοποιήσει. Μια διεργασία αποτελείται από λειτουργίες ενημέρωσης, οι οποίες με τη σειρά τους αντιστοιχούν στις στοιχειώδεις ενημερώσεις, όπως αυτές παρουσιάστηκαν στην ενότητα 2.5.2. Μέσω των λειτουργιών ενημέρωσης δίνεται η δυνατότητα για ενημέρωση των οντοτήτων.

Στη συνέχεια του κεφαλαίου αναπτύσσουμε τις ιδέες που οδήγησαν στην ανάπτυξη του ΜΔΕ. Μελετούμε την έννοια της διεργασίας ενημέρωσης και τις δυνατότητες που δίνει στο μοντέλο. Τέλος παρουσιάζουμε τους τύπους λειτουργιών ενημέρωσης και τη σχέση που έχουν με τις στοιχειώδεις ενημερώσεις.

3.1 Η προσέγγιση

Η αιτία για τη δημιουργία του ΜΔΕ, ήταν η ανάγκη διαλογικής ενημέρωσης στην εφαρμογή του ΣΣΕ, SIB Class Management System[2]. Εχοντας σαν σκοπό την ικανοποίηση μιας τέτοιας εφαρμογής αρχικά, προσπαθήσαμε να κατασκευάσουμε ένα γενικευμένο, όσο το δυνατό, μοντέλο διεργασιών (process model). Τα στάδια του μοντέλου, που αφορούν στην περιγραφή της διαδικασίας ενημέρωσης, αναλύονται σε τέτοιο βαθμό, ώστε να είναι εύκολα υλοποιήσιμα από το Δελτίο Εισαγωγής Δεδομένων [DT94].

Για το σχεδιασμό του μοντέλου, που περιγράφει τη διαδικασία διαλογικής εισαγωγής δεδομένων,

[1] Κύριος εισηγητής της κατασκευής του μοντέλου ήταν ο Martin Dörr.

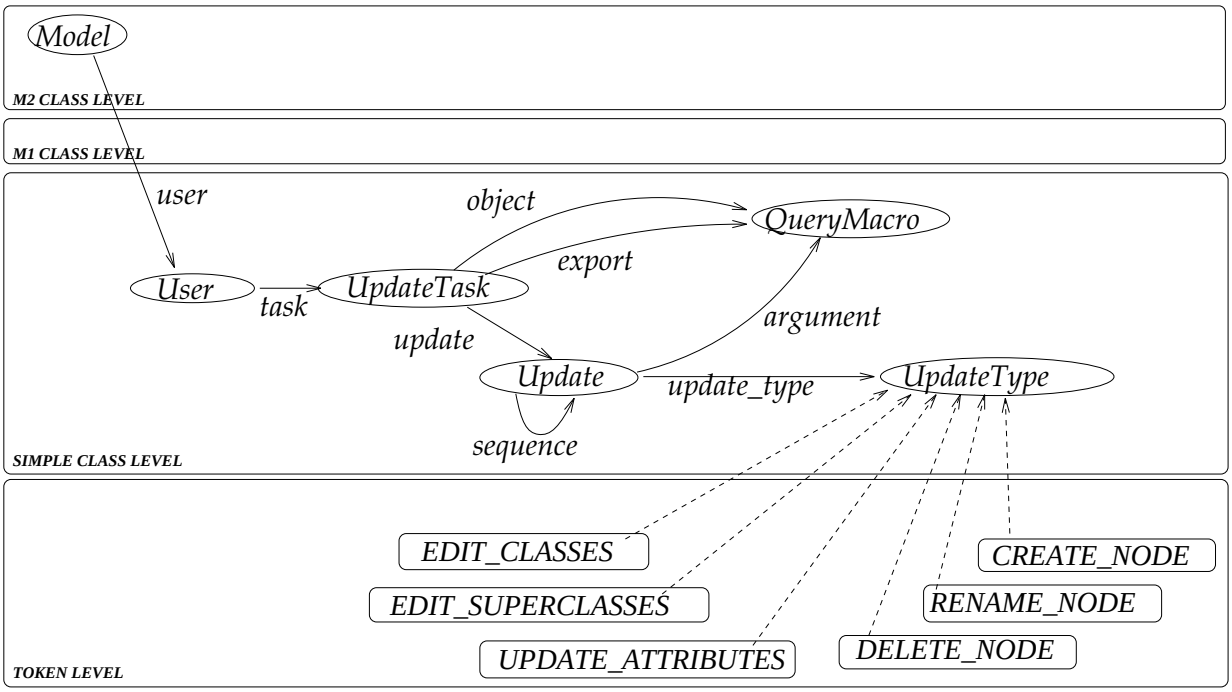
[2] Είναι μια εφαρμογή που αφορά στην τεκμηρίωση και την ανάλυση των στατικών ιδιοτήτων μονάδων λογισμικού.

βασιζόμαστε στην πεποίθηση ότι ένας χρήστης μπορεί να ενημερώσει τη βάση δεδομένων μέσω προκαθορισμένων διεργασιών ενημέρωσης (ΔΕ) (θα καλούνται και απλώς διεργασίες). Οι διεργασίες αυτές χαρακτηρίζουν το είδος της εργασίας ενημέρωσης που θέλουμε να εκτελεστεί. Περιορίζουν το χρήστη σε λειτουργίες που μπορεί να εκτελέσει και σε οντότητες που μπορεί να ενημερώσει, αποδίδοντας έτσι έννοιες δικαιοδοσιών ενημέρωσης. Σύμφωνα με το [Τζί95b], οι διεργασίες αποτελούν όψεις ενημέρωσης, ικανές να διαμερίσουν τη βάση σε ανεξάρτητα, ως προς την εξέλιξή τους, τμήματα.

Επιλέγοντας μια διεργασία, ο χρήστης μπορεί να διαλέξει ποιες οντότητες θα ενημερωθούν και με ποιες συγκεκριμένες λειτουργίες. Στις ΔΕ, μπορούμε να περιορίσουμε το σύνολο των οντοτήτων της βάσης, σε ένα σύνολο που μας δίνει τις πιθανές οντότητες προς ενημέρωση, για την επίτευξη του συγκεκριμένου σκοπού που εξυπηρετεί η διεργασία. Επίσης το σύνολο των προσφερομένων λειτουργιών μπορεί να περιοριστεί και να εξειδικευτεί σε λειτουργίες που αφορούν στη συγκεκριμένη διεργασία. Επομένως στις διεργασίες ομαδοποιούμε οντότητες και λειτουργίες που έχουν νόημα να βρίσκονται μαζί.

3.2 Γενική περιγραφή μοντέλου διεργασιών

Στις ιδέες που παρουσιάζονται στην ενότητα 3.1 βασίζεται το Μοντέλο Διεργασιών Ενημέρωσης, ή απλώς μοντέλο διεργασιών, που παριστάνεται γραφικά στο σχήμα 3.1.



Σχήμα 3.1: Σχηματική περιγραφή του Μοντέλου Διεργασιών Ενημέρωσης

Το μοντέλο της εφαρμογής την οποία καλούμαστε να ενημερώσουμε, συνδέεται με το ΜΔΕ μέσω

του ονόματος του μοντέλου εφαρμογής (Model στο σχήμα). Ετσι το μοντέλο εφαρμογής μπορεί να συνδέεται με έναν ή περισσότερους χρήστες (User). Ο χρήστης με τη σειρά του μπορεί να έχει στη διάθεσή του ένα σύνολο από διεργασίες ενημέρωσης (UpdateTask). Για να έχει δυνατότητα ενημέρωσης πρέπει να έχει τουλάχιστο μία διεργασία συνδεδεμένη με αυτόν.

Κάθε διεργασία πρέπει να έχει ένα σύνολο από προσφερόμενες λειτουργίες ενημέρωσης (Update) (θα καλούνται και απλώς λειτουργίες). Μέσω του γνωρίσματος *object* ορίζεται το σύνολο των οντοτήτων στο οποίο θα δράσουν οι λειτουργίες ενημέρωσης της διεργασίας. Το σύνολο αυτό, όπως κάθε άλλο σύνολο οντοτήτων στο ΜΔΕ, ορίζεται με ένα QueryMacro (δες ενότητα 3.3.1).

Προαιρετικό για τη διεργασία είναι το σύνολο *export*. Είναι αυτό που δίνει πληροφορίες απαραίτητες για την εξαγωγή πληροφορίας, για ενημερώσεις που έγιναν από τη διεργασία. Περιέχει το σύνολο των οντοτήτων που μπορούν να ενημερωθούν από τη διεργασία, καθώς και τις σχέσεις τους (περίπτωσης, εξειδίκευσης, απόδοσης γνωρισμάτων) προς άλλες οντότητες της βάσης[3].

Κάθε λειτουργία ενημέρωσης που περιγράφεται στο μοντέλο πρέπει να έχει ένα συγκεκριμένο τύπο (UpdateType), έναν από αυτούς που εμφανίζονται στο σχήμα 3.1. Επίσης μπορεί να έχει όρισμα (*argument*) που αποτελείται από ένα σύνολο οντοτήτων. Ανάλογα με τον τύπο της λειτουργίας ενημέρωσης, το σύνολο ορισμάτων έχει ξεχωριστή σημασία (δες ενότητα 3.5).

Προαιρετικά μια λειτουργία ενημέρωσης μπορεί να ακολουθείται από κάποια άλλη (μέσω του γνωρίσματος *sequence*). Ετσι μπορούμε να ορίσουμε ακολουθίες από λειτουργίες ενημέρωσης. Σκοπός είναι η διατήρηση περιορισμών ακεραιότητας, που επιτυγχάνεται με την ενημέρωση της βάσης, με περισσότερες από μία λειτουργίες, σε μια δοσοληψία.

3.3 Πραγματοποίηση μοντέλου διεργασιών

Για την παράσταση της διαδικασίας εισαγωγής δεδομένων σε κάθε εφαρμογή του ΣΣΕ, κατασκευάζεται από έναν ειδικό χρήστη, ένα μοντέλο, πραγματοποίηση του μοντέλου διεργασιών. Θα το καλούμε μοντέλο διεργασιών της εκάστοτε εφαρμογής ή, για απλότητα, μοντέλο διεργασιών και θα το διαχωρίζουμε από το -γενικό- μοντέλο διεργασιών, με βάση τα συμφραζόμενα.

Η πραγματοποίηση του μοντέλου αφορά στην κατασκευή διεργασιών που επιθυμούμε να εκτελούνται στο μοντέλο εφαρμογής. Μοντελοποιούνται οι χρήστες του συστήματος και αποδίδονται σε αυτούς διεργασίες. Οι τελευταίες προσαρμόζονται στους σκοπούς της εφαρμογής που πρόκειται να ενημερώσουν. Συγκεκριμένα καθορίζεται ένα σύνολο σχετικών με τη διεργασία οντοτήτων, ενώ οι λειτουργίες προσαρμόζονται στους σκοπούς της διεργασίας, για την ενημέρωση αυτών των οντοτήτων. Επομένως ο ειδικός χρήστης που κατασκευάζει το μοντέλο διεργασιών, πρέπει όχι μόνο να γνωρίζει τα περιεχόμενα της βάσης που η εφαρμογή διαχειρίζεται, αλλά να μπορεί να αποδίδει με

[3]Εχει υλοποιηθεί σχετικός μηχανισμός από το Γιάννη Τζίτζικα, που εξάγει τα δεδομένα μιας διεργασίας σε αρχεία με το συντακτικό της γλώσσας Telos [DT94].

κατανοητό τρόπο διεργασίες που στοχεύουν στην ενημέρωση του σχήματος του μοντέλου εφαρμογής και των δεδομένων του.

3.3.1 Καθορισμός συνόλων

Σημαντικός παράγοντας στην πραγματοποίηση του μοντέλου διεργασιών, αποτελεί ο καθορισμός συνόλων. Τα σύνολα ορίζουν το σύνολο των οντοτήτων στις οποίες οι διεργασίες θα δράσουν, ενώ επίσης εξειδικεύουν τις λειτουργίες. Στο ΜΔΕ, όπως το περιγράψαμε στην ενότητα 3.2, η παράσταση συνόλων γίνεται με τη βοήθεια των QueryMacro. Αυτά είναι ακολουθίες εντολών της γλώσσας ερωτήσεων (query interface - qi) του ΣΣΕ [DDP93]. Για να μπορέσει να ορίσει κάποιος σύνολα, πρέπει να γράψει ακολουθίες ερωτηματικών εντολών της γλώσσας του qi. Αυτό όμως είναι και ένα μειονέκτημα του καθορισμού συνόλων, αφού, ελλείψει μιας πιο φιλικής προς το χρήστη γλώσσας καθορισμού ερωτήσεων, ο κατασκευαστής του μοντέλου αναγκάζεται να χρησιμοποιεί το qi. Σχεδιάζεται ήδη, για το ΣΣΕ, μια άλλη γλώσσα ερωτήσεων, η TELQUEL, που υπόσχεται να είναι πιο φιλική προς το χρήστη. Πάντως για τον έμπειρο χρήστη του συστήματος και γνώστη της γλώσσας ερωτήσεων, είναι αρκετά εύκολο να ορίσει σύνολα μέσω των QueryMacro.

Το χαρακτηριστικό όλων των QueryMacro είναι ότι ξεκινούν με την εντολή καθορισμού εστιακής οντότητας του qi, *scn* (set current node), η οποία συνοδεύεται από το όνομα της εστιακής οντότητας. Ξεκινώντας από αυτήν την εστιακή οντότητα, μπορούμε να κάνουμε ένα πλήθος από ερωτήσεις στη γλώσσα του qi, καταλήγοντας στον καθορισμό ενός συνόλου οντοτήτων. Αποτέλεσμα κάθε QueryMacro είναι ένα σύνολο οντοτήτων της βάσης. Ενα παράδειγμα χρήσης τους φαίνεται στο σχήμα 3.2.

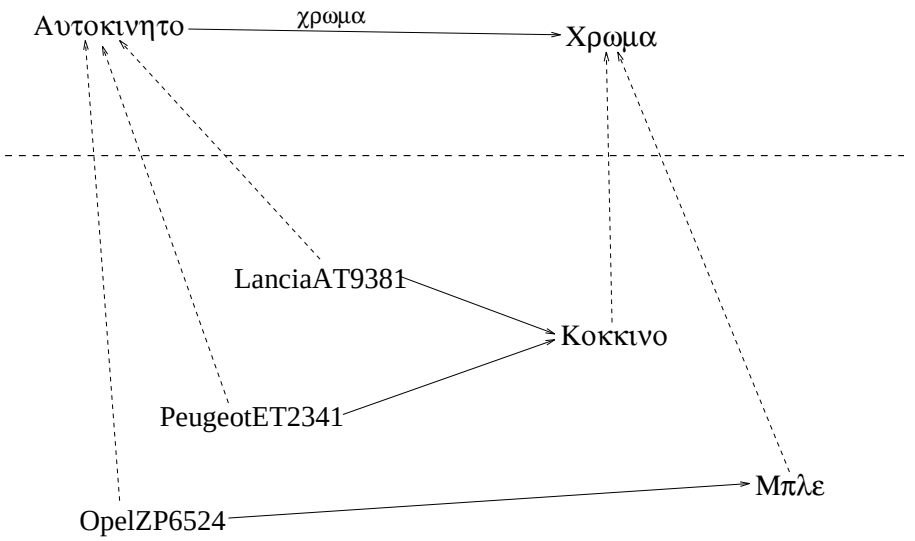
Τα QueryMacro μπορεί να είναι παραμετρικά. Η παράμετρος δίνεται σαν όρισμα στην εντολή καθορισμού εστιακής οντότητας (*scn*), κάτι που γίνεται κατά την κλήση του QueryMacro. Δίνεται η δυνατότητα να καθορίζουμε τον τύπο του ορίσματος που μπορεί το QueryMacro να δεχθεί (δες σχήμα 3.2), γεγονός που εκμεταλλευόμαστε στο ΔΕΔ, όπως θα δούμε σε επόμενο κεφάλαιο.

3.4 Διεργασίες ενημέρωσης

Η έννοια της διεργασίας ενημέρωσης έχει ιδιαίτερη σημασία για το ΜΔΕ. Οπως είδαμε αποτελεί το είδος της εργασίας που καλούμαστε να εκτελέσουμε. Τυπικά, στο μοντέλο διεργασιών μπορούμε να ορίσουμε μια ΔΕ με δύο σύνολα:

- το σύνολο των κόμβων που επιτρέπεται να ενημερωθούν από τις λειτουργίες της διεργασίας.
- το σύνολο των λειτουργιών που επιτρέπεται να εκτελεστούν στο παραπάνω σύνολο.

Ο ορισμός της ΔΕ με τον τρόπο που μόλις περιγράψαμε, εξυπηρετεί δύο βασικούς σκοπούς: Εκχώρηση δικαιοδοσιών και καθοδήγηση του χρήστη. Καθοριστικό ρόλο όμως για τη διεργασία, έχει το σύνολο



Σχήμα 3.2: Παράδειγμα χρήσης QueryMacro

Για τη βάση που φαίνεται στο σχήμα, δίνουμε δύο QueryMacro, ορισμένα στο συντακτικό της γλώσσας Telos.

Tell Cars in Token, QueryMacro with code
(1) : scn Αυτοκίνητο /* set current node */
(2) : gi /* get instances */
end

Δίνοντας σαν εστιακή οντότητα το **Αυτοκίνητο**, ζητάμε τις περιπτώσεις του (gi). Παίρνουμε απάντηση το σύνολο {LanciaAT9381, PeugeotET2341, OpelZP6524}

Tell Car'sColor in Token, QueryMacro with code
(1) : scn * Αυτοκίνητο /* set current node */
(2) : gtnc Αυτοκίνητο χρώμα /* get to node by category */
end

Το * δηλώνει παραμετρικό QueryMacro. Επιτρέπεται να δοθεί σαν παράμετρος ένας κόμβος, περίπτωση της κλάσης **Αυτοκίνητο**. Κάνοντας εστιακή οντότητα αυτόν τον κόμβο, ζητάμε την τιμή των γνωρισμάτων του για την κατηγορία (**Αυτοκίνητο**, *χρώμα*) (gtnc). Ετσι αν δώσουμε όρισμα το PeugeotET2341, παίρνουμε απάντηση το σύνολο {Κόκκινο}

των οντοτήτων στο οποίο αυτή ενεργεί.

3.4.1 Πεδίο δράσης διεργασίας

Οπως είδαμε στην ενότητα 3.2, μια διεργασία μπορεί να δράσει σε ένα σύνολο οντοτήτων, το οποίο ορίζεται με το γνώρισμα *object*. Το σύνολο αυτό καλούμε πεδίο δράσης διεργασίας (ΠΔΔ). Πρακτικοί λόγοι επέβαλαν να θεωρήσουμε ότι το ΠΔΔ αποτελείται από κόμβους. Ενας σοβαρός λόγος είναι το γεγονός ότι οι κόμβοι αποτελούν ανεξάρτητες οντότητες, που μπορεί κανείς να προσδιορίζει εύκολα. Αντίθετα τα γνωρίσματα είναι εξαρτημένα από τους κόμβους από τους οποίους εκκινούν. Για να προσδιορίσουμε ένα γνώρισμα πρέπει να αναφερθούμε πρώτα στον κόμβο εκκίνησης, κάτι που κάνει τον προσδιορισμό τους άμεσα εξαρτώμενο από τον προσδιορισμό του κόμβου εκκίνησης.

Εξάλλου στη διαλογική ενημέρωση, θέλουμε να βλέπουμε τα γνωρίσματα περισσότερο ως συνδέσμους παρά ως οντότητες. Αυτή είναι η διαισθητική αντίληψη που έχει ο χρήστης για αυτά. Το γεγονός ότι το ΣΣΕ τα μεταχειρίζεται ως οντότητες έχει σα συνέπεια να προσφέρονται επιπλέον δυνατότητες ενημέρωσης σε αυτά, διαφοροποιώντας τα σε κάποια σημεία από τους συνδέσμους

περίπτωσης και εξειδίκευσης[4]. Στο μέτρο του επιτρεπτού όμως, επιδιώκουμε, όπως θα φανεί και στη συνέχεια, να εμφανίζουμε μεγαλύτερη ομοιότητα μεταξύ των γνωρισμάτων και των συνδέσμων περίπτωσης και εξειδίκευσης, παρά μεταξύ των γνωρισμάτων και των κόμβων.

Με βάση τις παραπάνω παρατηρήσεις, μπορούμε να θεωρήσουμε οι οντότητες που η διεργασία έχει στη δικαιοδοσία της να ενημερώσει, προκύπτουν από τους κόμβους του πεδίου δράσης της και τα γνωρίσματα που εκκινούν από αυτούς[5]. Αν σε αυτές τις οντότητες προσθέσουμε τις σύνθετες οντότητες που, όπως θα δούμε στην ενότητα 6.1.2, αποτελούνται από κόμβους που δεν βρίσκονται υποχρεωτικά στο ΠΔΔ, έχουμε το σύνολο των οντοτήτων που *τελικά* μπορεί η διεργασία να ενημερώσει. Το σύνολο αυτό ονομάζουμε επεκτεταμένο πεδίο δράσης διεργασίας.

Στα πλαίσια της παρούσας εργασίας, θεωρήσαμε χρήσιμο να επιτρέψουμε διεργασίες χωρίς καθορισμένο από το μοντέλο ΠΔΔ. Δήλωση μιας τέτοιας ΔΕ, δίνει στο χρήστη τη δυνατότητα να μπορεί να εφαρμόζει τις προκαθορισμένες από το μοντέλο λειτουργίες της σε κάθε κόμβο της βάσης. Σε αυτή την περίπτωση το επεκτεταμένο πεδίο δράσης είναι όλες οι οντότητες της βάσης (κόμβοι και γνωρίσματα). Διεργασίες με τόσο διευρυμένο πεδίο δράσης αποσκοπούν στη δημιουργία σχήματος ή γενικότερα στην ενημέρωση οντοτήτων που δεν μπορούν να ενημερωθούν από άλλες εξειδικευμένες διεργασίες. Προφανώς ένα τέτοιο πεδίο δράσης καθιστά τη διεργασία πολύ γενική, με δυνατότητα ενημέρωσης κάθε κόμβου, και καλό είναι να απευθύνεται σε χρήστες με καλή γνώση των περιεχομένων της βάσης.

3.4.2 Εκχώρηση δικαιοδοσιών

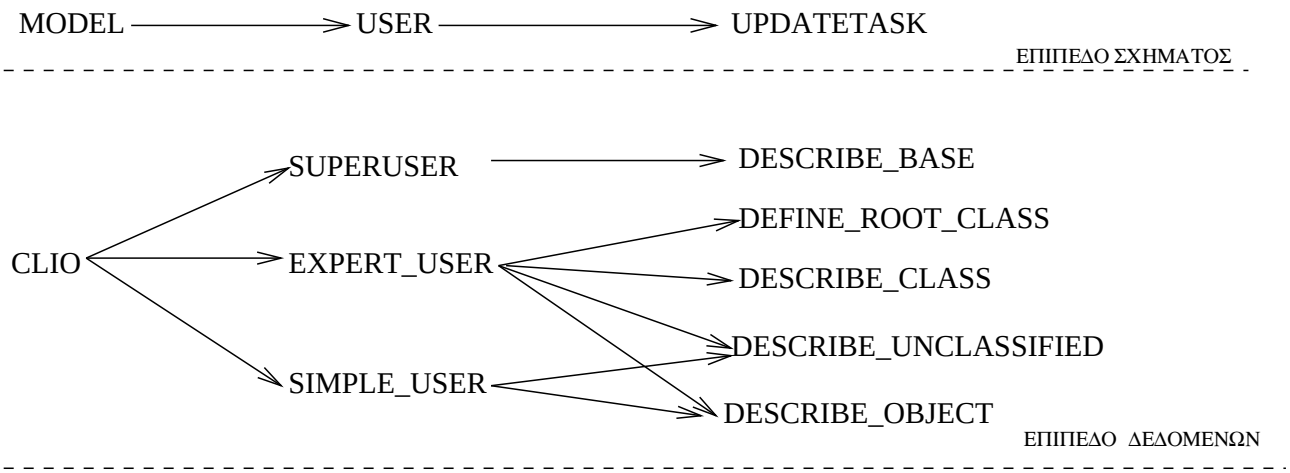
Στα πλαίσια της παρούσας εργασίας, θεωρήσαμε χρήσιμο να εκμεταλλευτούμε τις δυνατότητες του ΜΔΕ για εκχώρηση δικαιοδοσιών. Είναι καλό κάθε χρήστης να περιορίζεται σε ενέργειες της αρμοδιότητάς του που δε βλάπτουν την ακεραιότητα της βάσης. Αν ένα σύστημα ενημέρωσης δεν δίνει μηχανισμούς εκχώρησης δικαιοδοσιών, έχοντας όλους τους χρήστες στην ίδια στάθμη από πλευράς δικαιωμάτων ενημέρωσης, τότε είτε όλοι όσοι το χρησιμοποιούν περιορίζονται σε τετριμμένες σχετικά ενημερώσεις, σε μια προσπάθεια προφύλαξης δεδομένων από ανεπιθύμητες τροποποιήσεις, είτε επιτρέπονται πράξεις που δε θα ήταν φρόνιμο να γίνονται από ‘‘ακατάλληλους’’ χρήστες.

Με τις ΔΕ του μοντέλου, θεωρούμε ότι μπορούμε να αποδώσουμε δικαιοδοσίες. Κάθε ΔΕ που συνδέεται με ένα χρήστη, του επιτρέπει να δράσει σε σύνολα οντοτήτων με συγκεκριμένες λειτουργίες. Μπορούμε λοιπόν να δούμε κάθε διεργασία που του προσφέρεται, σαν δικαίωμα ενημέρωσης των οντοτήτων του επεκταμένου πεδίου δράσης της, μέσω των λειτουργιών που η διεργασία προσφέρει. Το σύνολο των ΔΕ για κάθε χρήστη αποτελεί μια έκφραση των δικαιωμάτων ενημέρωσης που αυτός έχει στη συγκεκριμένη βάση. Κάθε ένας έχει πρόσβαση σε διεργασίες που

[4]Τα γνωρίσματα μπορεί να έχουν όνομα, να ταξινομούνται και να εξειδικεύονται σε κατηγορίες γνωρισμάτων και να αποδίδονται σε αυτά γνωρίσματα, ιδιότητες που δεν έχουν οι σύνδεσμοι περίπτωσης και εξειδίκευσης.

[5]αρκεί να έχουν οριστεί για τη διεργασία λειτουργίες τύπου ενημέρωσης γνωρισμάτων

τον περιορίζουν σε συγκεκριμένες ενέργειες. Ετσι, μπορεί να γίνει εύκολα εκχώρηση δικαιοδοσιών, όπως φαίνεται και στο σχήμα 3.3.



Σχήμα 3.3: Εκχώρηση δικαιοδοσιών με το μοντέλο διεργασιών

Ο **SuperUser** έχει τη δυνατότητα να περιγράψει όλες τις οντότητες της βάσης μέσω της διεργασίας *DescribeBase*. Ο **ExpertUser** έχει τη δυνατότητα να περιγράψει τις κλάσεις της βάσης μέσω της διεργασίας *DescribeClass*. Επίσης μπορεί να καθορίσει σημεία ''εισόδου'' (RootClass) μέσω της διεργασίας *DefineRootClass*. Μπορεί να ταξινομήσει αταξινόμητες οντότητες μέσω της διεργασίας *DescribeUnclassifed*. Τέλος μπορεί να περιγράψει τις ατομικές οντότητες της βάσης μέσω της διεργασίας *DescribeObject*. Αυτές οι δύο τελευταίες διεργασίες είναι οι μόνες που αποδίδονται στον **SimpleUser** που έχει και τις λιγότερες δικαιοδοσίες.

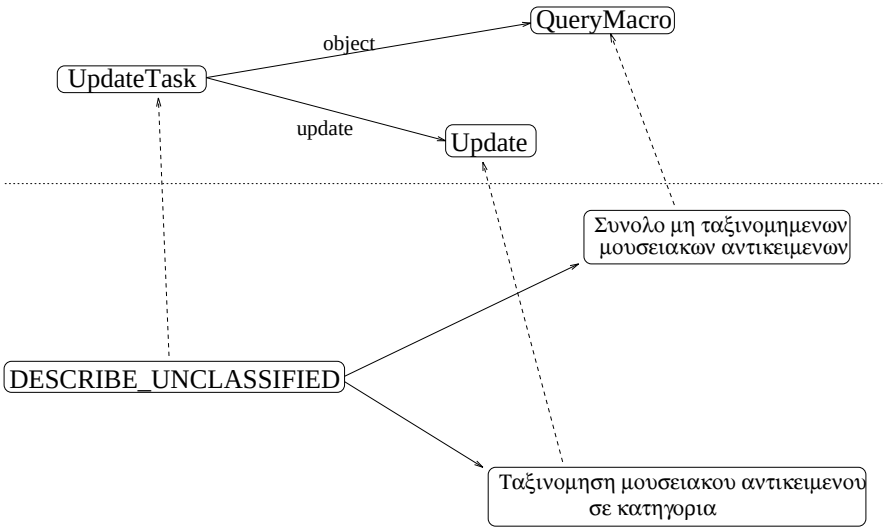
3.4.3 Καθοδήγηση του χρήστη

Ο χρήστης είναι καλό να μπορεί να επιλέγει από ένα μικρό σχετικά σύνολο τις οντότητες που μπορεί να ενημερώσει. Επίσης από ένα εξειδικευμένο σύνολο λειτουργιών να μπορεί να βρίσκει ποιες είναι οι λειτουργίες που έχει τη δυνατότητα να εκτελέσει στις επιλεγμένες οντότητες.

Οι ΔΕ μπορούν να δράσουν καθοδηγητικά προς όφελος του χρήστη. Ο τελευταίος, που δεν είναι απαραίτητο να γνωρίζει καλά τα περιεχόμενα της βάσης, μπορεί να χρειάζεται βοήθεια για να ξεκινήσει και να εκτελέσει τελικά μια λειτουργία ενημέρωσης. Ετσι του προσφέρονται διεργασίες που τον βοηθούν στην προσπάθειά του αυτή. Το σύνολο των οντοτήτων που είναι στο πεδίο δράσης κάθε διεργασίας και οι λειτουργίες που αυτή προσφέρει είναι ικανές να τον οδηγήσουν στην επιθυμητή τελική επιλογή. Παράδειγμα ΔΕ ως μηχανισμού καθοδήγησης φαίνεται στο σχήμα 3.4.

3.5 Λειτουργίες ενημέρωσης

Στην ενότητα 3.2 είδαμε ότι οι λειτουργίες ενημέρωσης όπως τις περιγράψαμε στο ΜΔΕ ανήκουν σε συγκεκριμένους τύπους. Οι τύποι αυτοί είναι : CREATE_NODE, DELETE_NODE, RENAME_NODE, EDIT_CLASSES, EDIT_SUPERCLASSES, UPDATE_ATTRIBUTES. Σε αυτούς τους τύπους αντιστοιχούν όλες οι στοιχειώδεις ενημερώσεις, που είδαμε στο προηγούμενο κεφάλαιο.



Σχήμα 3.4: Καθοδήγηση του χρήστη

Η διεργασία *DescribeUnclassifed* επιτρέπει την ταξινόμηση των μη ταξινομημένων μουσειακών αντικειμένων (το παράδειγμα είναι από την εφαρμογή του ΚΛΕΙΩ). Το πεδίο δράσης της είναι το σύνολο των μουσειακών αντικειμένων που χαρακτηρίζονται ''αταξινόμητα'' (Unclassified). Η μόνη λειτουργία που προσφέρεται στο χρήστη είναι η λειτουργία ταξινόμησης μουσειακού αντικειμένου σε κάποια από τις κατηγορίες μουσειακών ειδών. Το εξειδικευμένο πεδίο δράσης της και η μοναδική λειτουργία που προσφέρει, καθοδηγούν το χρήστη και δεν αφήνουν περιθώρια για λάθη.

Κάθε λειτουργία έχει μια οντότητα σαν εστιακή (θα την καλούμε εστιακή οντότητα (ΕΟ)), και είναι αυτή πάνω στην οποία θα δράσει. Η ΕΟ είναι κόμβος που επιλέγεται από το πεδίο δράσης της διεργασίας στην οποία κάθε λειτουργία ανήκει[6]. Ανάλογα με τον τύπο της, κάθε λειτουργία συνδέεται με διαφορετικές στοιχειώδεις ενημερώσεις, που τελικά ενημερώνουν την ΕΟ.

Το σύστημα σημασιολογικού ελέγχου του ΣΣΕ περιορίζει την ενημέρωση των οντοτήτων της βάσης με ένα σύνολο δομικών περιορισμών (δες ενότητα 2.5.1.2). Ετσι σε κάθε λειτουργία, αντιστοιχεί ένα σύνολο *σημασιολογικά ορθών* στοιχειωδών ενημερώσεων. Πολλές φορές όμως, για να ικανοποιήσουμε τους σκοπούς της διεργασίας στην οποία η λειτουργία ανήκει, αλλά και για λόγους περιορισμού και καθοδήγησης του χρήστη, θέλουμε να περιορίσουμε ακόμα περισσότερο τις δυνατότητες της λειτουργίας. Για το σκοπό αυτό χρησιμοποιούμε ένα σύνολο ορισμάτων (*argument* στο μοντέλο διεργασιών) που περιορίζει την εμβέλειά της, καθορίζοντας τα επιτρεπτά ορίσματα των στοιχειωδών ενημερώσεών της. Ανάλογα με τον τύπο της ΛΕ, το σύνολο αυτό έχει διαφορετική ερμηνεία, όπως θα δούμε στη συνέχεια. Ομως σε κάθε περίπτωση, η ύπαρξη του συνόλου αυτού περιορίζει τη λειτουργικότητα της ΛΕ.

Από την εφαρμογή του ΜΔΕ σε συγκεκριμένα συστήματα (όπως το ΚΛΕΙΩ), είδαμε πως έχει νόημα να θεωρήσουμε λειτουργίες που δεν έχουν σύνολο ορισμάτων το οποίο να περιορίζει την εμβέλειά τους. Ετσι στα πλαίσια της παρούσας εργασίας, επιτρέψαμε τον ορισμό τέτοιων λειτουργιών στο

[6]Εξαίρεση αποτελεί η λειτουργία τύπου CREATE_NODE, για την οποία η ΕΟ είναι ο κόμβος που πρόκειται να δημιουργηθεί. Αυτός δεν ανήκει στο ΠΔΔ, αφού δεν υπάρχει ακόμα στη βάση.

ΜΔΕ. Λειτουργία χωρίς σύνολο ορισμάτων, μπορεί να ενημερώσει την ΕΟ με όλες τις σημασιολογικά ορθές στοιχειώδεις ενημερώσεις που αντιστοιχούν στο τύπο της, χωρίς περιορισμούς. Απευθύνονται σε χρήστες που κρίνονται από τον κατασκευαστή του μοντέλου ικανοί ότι μπορούν να ενημερώσουν την ΕΟ με τη συγκεκριμένη λειτουργία, χωρίς να προκληθούν δυσάρεστες καταστάσεις στη βάση.

Ας δούμε πως συνδέονται οι τύποι λειτουργιών ενημέρωσης με τις στοιχειώδεις ενημερώσεις και τι νόημα έχει σε κάθε περίπτωση το όρισμα της λειτουργίας, αν υπάρχει. Λειτουργία με τύπο:

1. CREATE_NODE :

   Υλοποιεί τη στοιχειώδη ενημέρωση *δημιουργία κόμβου* (*λογικό όνομα*, *επίπεδο ταξινόμησης*).
   Το σύνολο ορισμάτων της λειτουργίας

   - αν υπάρχει, μας δίνει τις κλάσεις στις οποίες θα γίνει περίπτωση ο κόμβος που πρόκειται να δημιουργηθεί. Ετσι υλοποιείται και η στοιχειώδης ενημέρωση *δημιουργία συνδέσμου περίπτωσης* (*οντότηταA*, *οντότηταB*), όπου *οντότηταA* είναι ο κόμβος που πρόκειται να δημιουργηθεί, ενώ *οντότηταB* είναι κάθε κλάση του ορίσματος της λειτουργίας. Το *επίπεδο ταξινόμησης* συμπεραίνεται αυτόματα σε αυτή την περίπτωση.
   - αν δεν υπάρχει, τότε οποιοσδήποτε κόμβος που υπάρχει ήδη στη βάση μπορεί να είναι η κλάση στην οποία θα γίνει περίπτωση ο κόμβος που πρόκειται να δημιουργηθεί. Αν αυτή η κλάση είναι καθορισμένη από το χρήστη, υλοποιείται η στοιχειώδης ενημέρωση *δημιουργία συνδέσμου περίπτωσης* (*οντότηταA*, *οντότηταB*), όπου *οντότηταA* είναι ο κόμβος που πρόκειται να δημιουργηθεί, ενώ *οντότηταB* είναι η οντότητα στην οποία αυτός θα γίνει περίπτωση. Το *επίπεδο ταξινόμησης* συμπεραίνεται αυτόματα.
     Αν αντί για κλάση καθορισμένη από το χρήστη, δοθεί το *επίπεδο ταξινόμησης*, τότε απλώς η στοιχειώδης ενημέρωση *δημιουργία συνδέσμου περίπτωσης* δεν υλοποιείται.

2. DELETE_NODE :

   Υλοποιεί τη στοιχειώδη ενημέρωση *διαγραφή κόμβου* (*λογικό όνομα*).
   Το *λογικό όνομα* είναι το όνομα της ΕΟ της λειτουργίας. Το σύνολο ορισμάτων της λειτουργίας

   - αν υπάρχει, δίνει το σύνολο των κόμβων του πεδίου δράσης της διεργασίας, που επιτρέπεται να διαγραφούν.
   - αν δεν υπάρχει, όλοι οι κόμβοι του πεδίου δράσης της διεργασίας επιτρέπεται να διαγραφούν.

3. RENAME_NODE :

   Υλοποιεί τη στοιχειώδη ενημέρωση *μετονομασία οντότητας* (*οντότητα*, *λογικό όνομα*), μόνο όμως για την περίπτωση που η *οντότητα* είναι κόμβος.
   Σαν *οντότητα* θεωρούμε την ΕΟ, που το όνομά της αλλάζει σε *λογικό όνομα*. Το σύνολο ορισμάτων της λειτουργίας

- αν υπάρχει, δίνει το σύνολο των κόμβων του πεδίου δράσης της διεργασίας, που επιτρέπεται να μετονομαστούν.
- αν δεν υπάρχει, όλοι οι κόμβοι του πεδίου δράσης της διεργασίας επιτρέπεται να μετονομαστούν.

4. EDIT_CLASSES :

   Υλοποιεί στοιχειώδεις ενημερώσεις δημιουργίας και διαγραφής συνδέσμων περίπτωσης, μόνο όμως μεταξύ κόμβων. Συγκεκριμένα

   - αν πρόκειται για δημιουργία, υλοποιεί στοιχειώδεις ενημερώσεις *δημιουργίας συνδέσμου περίπτωσης* (*οντότηταA*, *οντότηταB*). Η ΕΟ γίνεται *οντότηταA*. Η *οντότηταB* βρίσκεται σε συνάρτηση με τις οντότητες του συνόλου ορισμάτων της λειτουργίας, αν αυτό υπάρχει. Ετσι
     -- αν το σύνολο ορισμάτων της λειτουργίας υπάρχει, περιέχει τους κόμβους που μπορούν να αποτελέσουν πιθανές κλάσεις για την ΕΟ. Η *οντότηταB* είναι κάποιο από τα στοιχεία του συνόλου.
     -- αν το σύνολο ορισμάτων της λειτουργίας δεν υπάρχει, τότε η *οντότηταB* μπορεί να είναι οποιοσδήποτε κόμβος στον οποίο η ΕΟ μπορεί να γίνει περίπτωση, σύμφωνα με το σύστημα σημασιολογικού ελέγχου του ΣΣΕ.
   - αν πρόκειται για διαγραφή, υλοποιεί στοιχειώδεις ενημερώσεις *διαγραφής συνδέσμου περίπτωσης* (*οντότηταA*, *οντότηταB*). Η ΕΟ γίνεται *οντότηταA*. Η *οντότηταB* βρίσκεται σε συνάρτηση με τις οντότητες του συνόλου ορισμάτων της λειτουργίας, αν αυτό υπάρχει. Ετσι
     -- αν το σύνολο ορισμάτων της λειτουργίας υπάρχει, περιέχει τις κλάσεις από τις οποίες επιτρέπεται να διαγράψουμε την ΕΟ. Η *οντότηταB* είναι κάποιο από τα στοιχεία του συνόλου.
     -- αν το σύνολο ορισμάτων της λειτουργίας δεν υπάρχει, τότε η *οντότηταB* μπορεί να είναι οποιοσδήποτε κόμβος στον οποίο η ΕΟ είναι ήδη περίπτωση. Δηλαδή δίνεται η δυνατότητα να κάνουμε όλες τις σημασιολογικά επιτρεπτές διαγραφές συνδέσμων περίπτωσης.

5. EDIT_SUPERCLASSES :

   Υλοποιεί στοιχειώδεις ενημερώσεις δημιουργίας και διαγραφής συνδέσμων εξειδίκευσης, μόνο όμως μεταξύ κόμβων. Εχει ακριβώς τις ίδιες δυνατότητες με λειτουργίες τύπου EDIT_CLASSES, μόνο που αντί για συνδέσμους περίπτωσης ενημερώνει συνδέσμους εξειδίκευσης.

6. UPDATE_ATTRIBUTES :

   Υλοποιεί στοιχειώδεις ενημερώσεις δημιουργίας, διαγραφής, μετονομασίας, ταξινόμησης,

εξειδίκευσης και απόδοσης γνωρίσματος, για τα γνωρίσματα της εστιακής οντότητας. Πιο συγκεκριμένα, οι ενημερώσεις γίνονται για τα γνωρίσματα των κατηγοριών που δίνονται από το όρισμα της λειτουργίας. Αυτές αποτελούν τις ενημερώσιμες κατηγορίες της ΕΟ. Αν δεν υπάρχει όρισμα, ο υπολογισμός των ενημερώσιμων κατηγοριών γίνεται ως εξής: Παίρνουμε τις κλάσεις της ΕΟ[7]. Ολα τα γνωρίσματα που εκκινούν από αυτές αποτελούν το σύνολο των κατηγοριών προς ενημέρωση.

Μπορούμε διαλογικά να επιλέξουμε ενημερώσιμη κατηγορία και συγκεκριμένο γνώρισμα σε αυτήν, οπότε η λειτουργία υλοποιεί τις παρακάτω ενημερώσεις :

- αν πρόκειται για δημιουργία γνωρίσματος υλοποιεί τις στοιχειώδεις ενημέρωσεις *δημιουργία γνωρίσματος* (*λογικό όνομα*, *επίπεδο ταξινόμησης*, *οντότητα εκκίνησης*, *τιμή γνωρίσματος*), για τα γνωρίσματα των ενημερώσιμων κατηγοριών. Συνδέεται η ΕΟ της λειτουργίας (είναι η *οντότητα εκκίνησης*), με την *τιμή γνωρίσματος*, που δίνεται διαλογικά. Για κάθε τέτοια ενημέρωση, υλοποιεί επίσης τη στοιχειώδη ενημέρωση *δημιουργία συνδέσμου περίπτωσης* μεταξύ του γνωρίσματος που δημιουργούμε και της ενημερώσιμης κατηγορίας που διαλογικά επιλέγουμε. Το *επίπεδο ταξινόμησης* του γνωρίσματος υπολογίζεται αυτόματα, εκτός αν κάτι τέτοιο δεν είναι δυνατό οπότε και ζητείται.
- αν πρόκειται για διαγραφή γνωρίσματος υλοποιεί τις στοιχειώδεις ενημέρωσεις *διαγραφή γνωρίσματος* (*λογικό όνομα*, *οντότητα εκκίνησης*, *τιμή γνωρίσματος*), για τα γνωρίσματα των ενημερώσιμων κατηγοριών. Διαγράφει το επιλεγμένο γνώρισμα που εκκινεί από την ΕΟ (είναι η *οντότητα εκκίνησης*), έχει *τιμή γνωρίσματος* την τιμή του *λογικό όνομα* το λογικό όνομα και βρίσκεται στην ενημερώσιμη κατηγορία που έχει επιλεγεί.
- αν πρόκειται για μετονομασία γνωρίσματος υλοποιεί τις στοιχειώδεις ενημέρωσεις *μετονομασία οντότητας* (*οντότητα*, *λογικό όνομα*), για τα γνωρίσματα των ενημερώσιμων κατηγοριών. Μετονομάζεται σε *λογικό όνομα* το όνομα του γνωρίσματος της ΕΟ που επιλέγεται διαλογικά.
- αν πρόκειται για δημιουργία/διαγραφή συνδέσμων περίπτωσης υλοποιεί τις στοιχειώδεις ενημερώσεις *δημιουργία/διαγραφή συνδέσμου περίπτωσης* (*οντότηταA*, *οντότηταB*), για τα γνωρίσματα των ενημερώσιμων κατηγοριών. Η *οντότηταA*, το γνώρισμα στο οποίο γίνεται δημιουργία/διαγραφή συνδέσμου περίπτωσης, αλλά και η *οντότηταB*, η κατηγορία στην οποία ταξινομείται/διαγράφεται, επιλέγονται διαλογικά.
- αν πρόκειται για δημιουργία/διαγραφή συνδέσμων εξειδίκευσης υλοποιεί τις στοιχειώδεις ενημερώσεις *δημιουργία/διαγραφή συνδέσμου εξειδίκευσης* (*οντότηταA*, *οντότηταB*), για τα γνωρίσματα των ενημερώσιμων κατηγοριών. Η *οντότηταA*, το γνώρισμα στο οποίο γίνεται

[7]για την ακρίβεια παίρνουμε όλες τις κλάσεις, άμεσες και έμμεσες

δημιουργία/διαγραφή συνδέσμου εξειδίκευσης, αλλά και η *οντότηταB*, η κατηγορία στην οποία γενικεύεται/διαγράφεται, επιλέγονται διαλογικά.

- αν πρόκειται για απόδοση γνωρίσματος, υλοποιεί τις στοιχειώδεις ενημερώσεις που υλοποιεί η λειτουργία UPDATE_ATTRIBUTES, για τα γνωρίσματα των ενημερώσιμων κατηγοριών. Η διαφορά είναι ότι εστιακή οντότητα της UPDATE_ATTRIBUTES γίνεται αυτή τη φορά το γνώρισμα που επιλέγεται διαλογικά.

Κεφάλαιο 4

Το Δελτίο Εισαγωγής Δεδομένων - Περιγραφή

Για τις ανάγκες του ΣΣΕ και ειδικά για τη διαλογική ενημέρωση των βάσεων δεδομένων που αυτό υποστηρίζει, έχουμε σχεδιάσει και υλοποιήσει ένα εργαλείο διαλογικής ενημέρωσης δεδομένων που ονομάζουμε Δελτίο Εισαγωγής Δεδομένων (ΔΕΔ). Οι απαιτήσεις που έχουμε από ένα τέτοιο εργαλείο είναι να:

- Δίνει πλήρες σύνολο πράξεων για ενημέρωση μιας βάσης.
- Παρέχει στο χρήστη αυτοματισμούς, διευκολύνσεις και μηχανισμούς καθοδήγησης κατά τη διάρκεια της διαλογικής ενημέρωσης.
- Προσαρμόζεται εύκολα ανάλογα με την εφαρμογή που καλείται να εξυπηρετήσει.
- Προσφέρει μηχανισμούς για καθορισμό δικαιοδοσιών ανάλογα με το χρήστη του συστήματος.
- Ελέγχει και διατηρεί την ακεραιότητα της βάσης.

Με το ΔΕΔ έχουμε πετύχει τους παραπάνω στόχους σε ικανοποιητικό βαθμό, όπως θα φανεί στη συνέχεια της εργασίας.

Το εργαλείο ακολουθεί το μοντέλο διεργασιών που περιγράψαμε στο 3ο κεφάλαιο. Μπορούν να γραφούν προδιαγραφές στο μοντέλο διεργασιών, τις οποίες ερμηνεύει κατάλληλα το ΔΕΔ και προσαρμόζει ανάλογα με αυτές τη λειτουργικότητά του. Εγινε προσπάθεια να φτιαχτεί ένα εργαλείο αρκετά ευέλικτο, ώστε να καλύπτει με την επαφή χρήσης του εφαρμογές των οποίων οι απαιτήσεις για εισαγωγή δεδομένων δεν είναι κοινές. Βέβαια ο πυρήνας του εργαλείου, το στοιχείο εκείνο που καθορίζει σχεδόν εξολοκλήρου τη συμπεριφορά του, είναι οι προδιαγραφές που παίρνει από το μοντέλο διεργασιών.

Το ΔΕΔ αποτελεί αυτόνομο εργαλείο, το οποίο μπορεί κανείς να χρησιμοποιήσει ανεξάρτητα από τα άλλα εργαλεία του ΣΣΕ. Ομως ο σχεδιασμός του έχει γίνει με το σκεπτικό ότι θα εμφανίζεται

παράλληλα με το γραφικό αναλυτή του ΣΣΕ. Αυτό γιατί με το γραφικό αναλυτή έχει κανείς τη δυνατότητα να βλέπει με διάφορους τρόπους τα περιεχόμενα της βάσης και τις αλλαγές που κάνει σε αυτήν με το ΔΕΔ. Για αυτό το λόγο, το τελευταίο δεν έχει τη μορφή μιας ολοκληρωμένης εφαρμογής με ένα κύριο παράθυρο και κάποια βοηθητικά, αλλά έχει τη μορφή καρτών που διαδέχονται η μια την άλλη, καταλήγοντας πάντα σε κάρτες ενημέρωσης. Η κύρια εφαρμογή θεωρούμε ότι είναι ο γραφικός αναλυτής, αφού συνήθως αυτός είναι που καλεί το ΔΕΔ. Εξάλλου η ακολουθία καρτών του εργαλείου που θα δούμε στη συνέχεια (επιλογή διεργασίας, επιλογή λειτουργίας και τελικά ενημέρωση οντότητας), απεικονίζει τη σειρά ενεργειών που επιβάλλονται από το μοντέλο διεργασιών και τελικά πρέπει να ακολουθήσει ο χρήστης.

Στη συνέχεια του κεφαλαίου παρουσιάζεται το ΔΕΔ με παραδείγματα χρήσης του από εφαρμογές του ΣΣΕ. Εξηγούμε τον τρόπο με τον οποίο ενεργεί ο χρήστης για την παρουσίαση των καρτών ενημέρωσης, και παρουσιάζουμε τη λειτουργικότητά τους.

4.1 Ροή ελέγχου

Για να εμφανίσουμε ενημερώσιμη πληροφορία στο ΔΕΔ, πρέπει να κάνουμε τα βήματα που επιβάλλονται από το μοντέλο διεργασιών, δηλαδή α) καθορισμός διεργασίας, β) επιλογή οντότητας προς ενημέρωση και καθορισμός λειτουργίας ενημέρωσης και γ) ενημέρωση της οντότητας μέσω της λειτουργίας. Τα βήματα αυτά αντιστοιχούν σε κάρτες. Ακολουθείται η ροή ελέγχου που φαίνεται στο διάγραμμα του σχήματος 4.1.

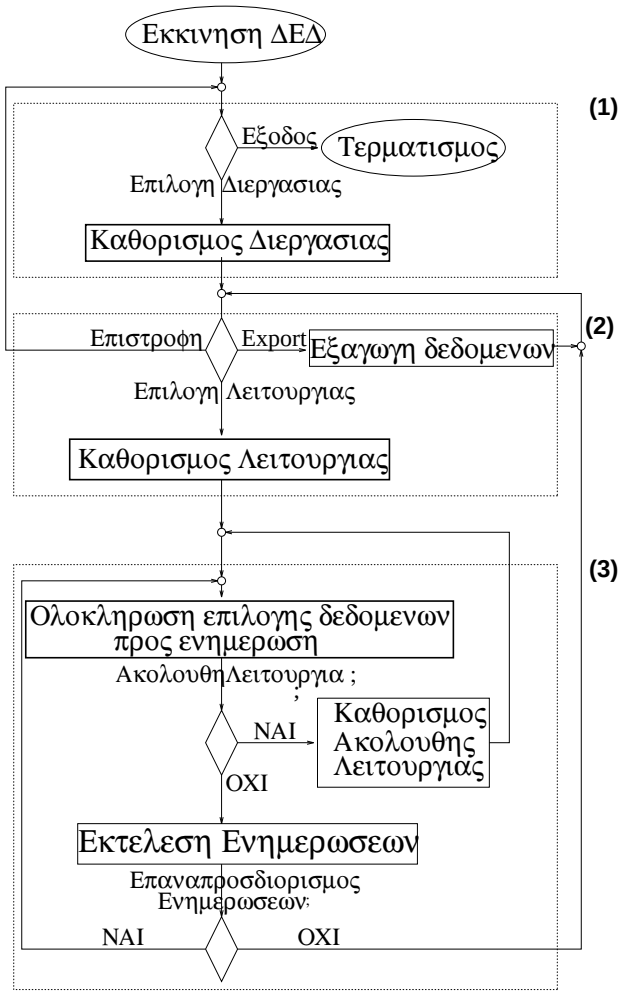
Αρχικά, ο χρήστης επιλέγει διεργασία ενημέρωσης από την πρώτη κάρτα του εργαλείου, την κάρτα επιλογής διεργασίας. Οι διεργασίες που προσφέρονται στο χρήστη εξαρτώνται από τις δικαιοδοσίες που του έχουν αποδοθεί στο μοντέλο διεργασιών (δες ενότητα 3.4.2).

Στη συνέχεια, από τη δεύτερη κάρτα, την κάρτα επιλογής λειτουργίας, καθορίζει τη λειτουργία που θα εκτελέσει και τον κόμβο που θα ενημερωθεί. Οι επιλογές που αυτή η κάρτα προσφέρει, εξαρτώνται από τη διεργασία που έχει καθορίσει ο χρήστης στην κάρτα επιλογής διεργασίας, αλλά και τον ορισμό της διεργασίας στο μοντέλο.

Ακολούθως εμφανίζεται κάρτα ενημέρωσης, αντίστοιχη του τύπου λειτουργίας που έχει επιλεγεί. Από αυτήν επιλέγει τα δεδομένα που θα αποτελέσουν τη νέα πληροφορία της βάσης.

Αν από το μοντέλο διεργασιών, για την λειτουργία που ο χρήστης έχει επιλέξει, έχει καθοριστεί ακόλουθη λειτουργία[1], εμφανίζεται νέα κάρτα ενημέρωσης. Οδηγούμαστε σε σύνθετη λειτουργία που είναι προκαθορισμένη από το μοντέλο διεργασιών. Τέλος, αφού οριστούν τα νέα δεδομένα, ο χρήστης καλείται να ζητήσει την εκτέλεση των ενημερώσεων.

[1]Με αυτόν τον όρο εννοούμε ένα σύνολο λειτουργιών ενημέρωσης που περνούν στη βάση σε μία δοσοληψία.



Σχήμα 4.1: Διάγραμμα ροής ελέγχου

Τα διακεκομμένα πλαίσια, τα οποία αριθμούνται, δίνουν τη σειρά εμφάνισης των καρτών του ΔΕΔ. Το (1) αντιστοιχεί στην κάρτα επιλογής διεργασίας, το (2) στην κάρτα επιλογής λειτουργίας και το (3) σε κάρτες ενημέρωσης. Βλέπουμε έτσι τις δυνατότητες που δίνει κάθε κάρτα του εργαλείου.

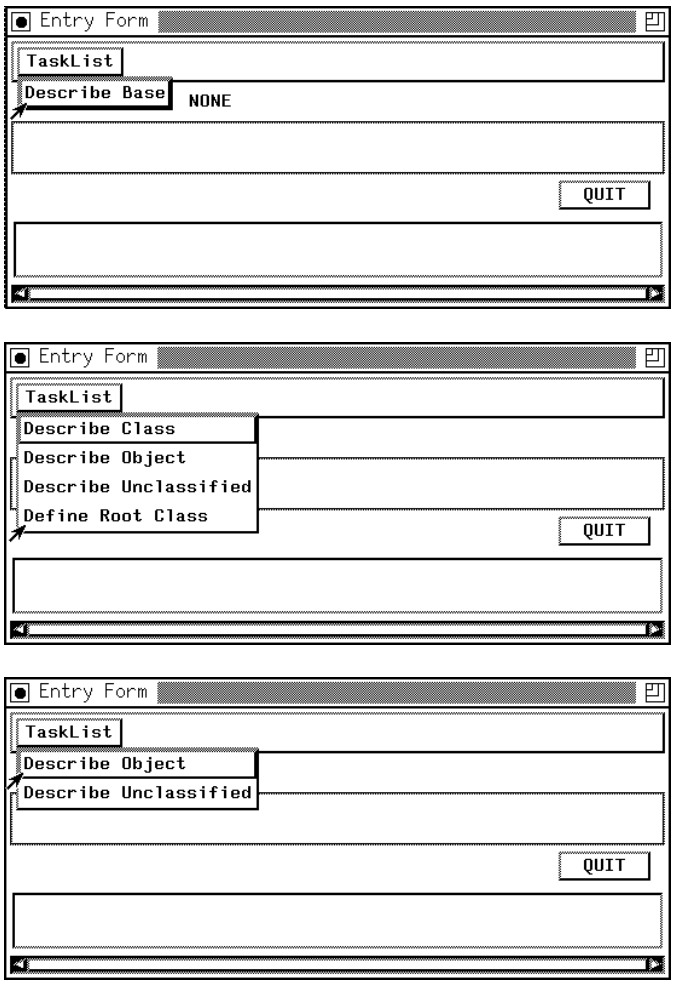
4.2 Εκκίνηση

Το εκτελέσιμο πρόγραμμα του ΔΕΔ καλείται με παράμετρο το μοντέλο της εφαρμογής (Model στο μοντέλο διεργασιών), την οποία θέλουμε να ενημερώσουμε. Επίσης, προαιρετικά, μπορούμε να δώσουμε κι ένα δεύτερο όρισμα που είναι το όνομα του χρήστη, ο οποίος θα χρησιμοποιήσει το σύστημα. Με αυτόν τον τρόπο μπορούμε να καθορίσουμε εξουσιοδοτήσεις, αν βέβαια στο μοντέλο διεργασιών έχουμε ορίσει τους χρήστες και τις δικαιοδοσίες τους (δες ενότητα 3.2). Αν δεν καθοριστεί χρήστης του συστήματος κατά την εκκίνηση, θεωρούμε ότι υπάρχει ένας και μοναδικός, με εξουσιοδοτήσεις αυτές που περιγράφονται στο μοντέλο.

Δίνοντας λοιπόν αυτές τις παραμέτρους ξεκινά το ΔΕΔ, εμφανίζοντας την πρώτη κάρτα στην οποία καλούμαστε να επιλέξουμε διεργασία ενημέρωσης.

4.3 Καθορισμός διεργασίας

Το ΔΕΔ προσφέρει στο χρήστη τις ΔΕ για τις οποίες έχει εξουσιοδοτηθεί από το μοντέλο διεργασιών. Παραδείγματα καθορισμού διεργασιών ανάλογα με το χρήστη του συστήματος έχουμε στο σχήμα 4.2. Παρουσιάζεται το αποτέλεσμα της μοντελοποίησης τριών χρηστών για την εφαρμογή του ΚΛΕΙΩ, σύμφωνα με το μοντέλο διεργασιών του σχήματος 3.3. Οι διεργασίες προσφέρονται στο χρήστη με τη μορφή λίστας επιλογών.



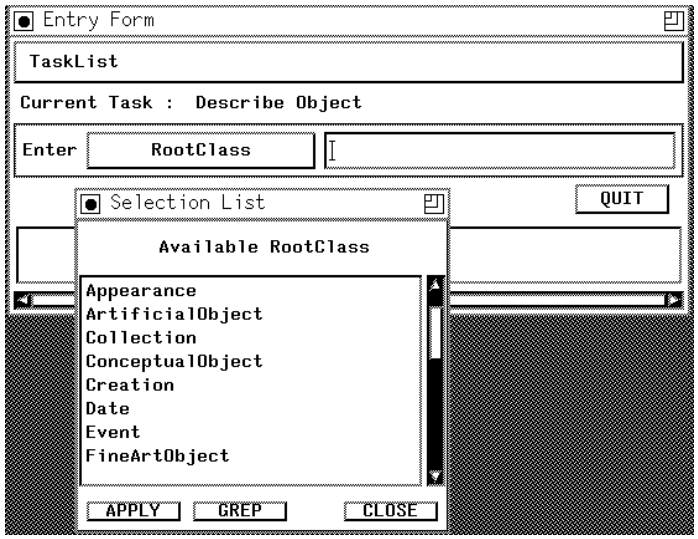
Σχήμα 4.2: Καθορισμός διεργασιών ανά χρήστη

Για το μοντέλο του ΚΛΕΙΩ, βλέπουμε διαφοροποιήσεις στις προσφερόμενες διεργασίες του ΔΕΔ, ανάλογα με το χρήστη που το χρησιμοποιεί. Η πρώτη αφορά το SuperUser, η δεύτερη τον ExpertUser, η τρίτη το SimpleUser, όπως αυτοί έχουν παρασταθεί στο μοντέλο διεργασιών. Η επιλογή της διεργασίας γίνεται από ένα μενού επιλογών (**TaskList**)

Οι ΔΕ, όπως είδαμε στο 3ο κεφάλαιο, μπορεί να έχουν ορισμένο το πεδίο δράσης τους με ένα Query Macro.

- Αν το QueryMacro που ορίζει το πεδίο δράσης της διεργασίας δεν υπάρχει, ή υπάρχει και δεν είναι παραμετρικό, τότε ο καθορισμός της διεργασίας ολοκληρώνεται μετά την επιλογή της από το μενού επιλογών.

- Αν το QueryMacro υπάρχει και είναι παραμετρικό, τότε η διεργασία για να καθοριστεί πλήρως πρέπει να πάρει ένα όρισμα από το χρήστη. Το όρισμα χρειάζεται για να καθοριστεί ακριβώς το πεδίο δράσης της διεργασίας. Για το σκοπό αυτό παρουσιάζεται ένα πεδίο στο οποίο μπορεί να πληκτρολογήσει το όρισμα. Εναλλακτικά μπορεί να επιλέξει την παρουσίαση μιας λίστας με τα πιθανά ορίσματα και να διαλέξει κάποιο από αυτά, όπως βλέπουμε και στο σχήμα 4.3. Τα περιεχόμενα της λίστας είναι συνάρτηση του τύπου του ορίσματος που περιμένουμε, στοιχείο που μας δίνεται από το παραμετρικό QueryMacro (δες ενότητα 3.3.1). Μετά τον καθορισμό της παραμέτρου, η διεργασία έχει καθοριστεί.



Σχήμα 4.3: Describe Object: Καθορισμός παραμετρικής διεργασίας

Για να καθοριστεί πλήρως η διεργασία **Describe Object**, πρέπει να οριστεί μια παράμετρος. Στην προκειμένη περίπτωση απαιτείται μια *RootClass* - είναι η κλάση που περιέχει τις κορυφές ιεραρχιών που ο κατασκευαστής του μοντέλου διεργασιών θεωρεί σημεία ''εισόδου'' στο μοντέλο εφαρμογής (το παράδειγμα είναι από το ΚΛΕΙΩ). Καλείται ο χρήστης να πληκτρολογήσει το όρισμα στο κατάλληλο πεδίο, ενώ εναλλακτικά μπορεί να ζητήσει μια λίστα με όλες τις RootClass, για επιλογή.

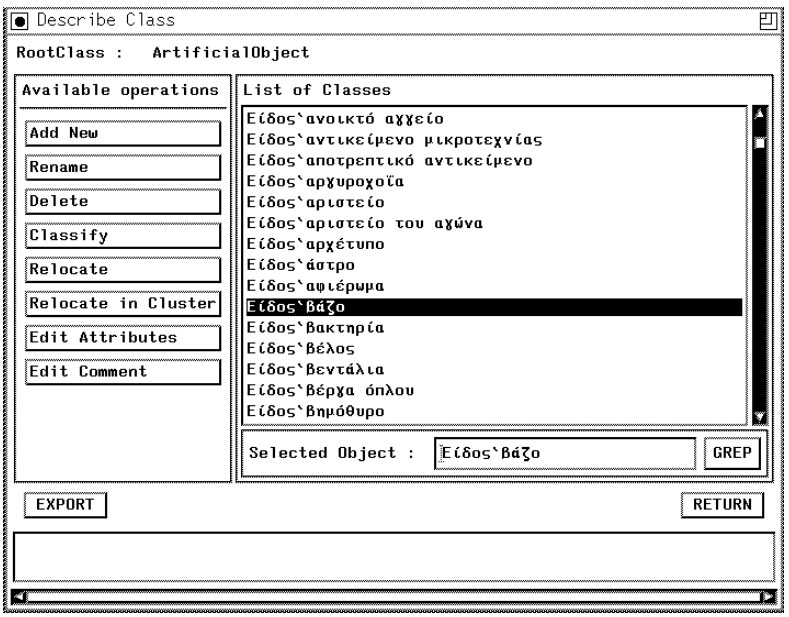
4.4 Καθορισμός λειτουργίας

Ο καθορισμός της λειτουργίας και της οντότητας στην οποία αυτή θα δράσει, είναι επιλογή του χρήστη. Ομως επειδή τα γνωρίσματα είναι εξαρτημένες οντότητες, ενώ οι κόμβοι ανεξάρτητες, έχουμε διαφορετικό τρόπο προσδιορισμού τους. Στη συνέχεια εξηγούμε πως γίνεται ο καθορισμός λειτουργιών σε κόμβους και πως σε γνωρίσματα.

4.4.1 Καθορισμός λειτουργίας για κόμβους

Η επιλογή λειτουργίας για κόμβους γίνεται από τη δεύτερη κάρτα του ΔΕΔ, όπως αυτή απεικονίζεται στο διάγραμμα ροής ελέγχου (σχήμα 4.1). Καλούμαστε να διαλέξουμε ένα κόμβο που βρίσκεται

στο πεδίο δράσης της διεργασίας και τον οποίο θέλουμε να ενημερώσουμε με κάποια από τις προσφερόμενες λειτουργίες. Το πεδίο δράσης της διεργασίας παρουσιάζεται με τη μορφή λίστας κόμβων. Οπως φαίνεται και στο σχήμα 4.4, ο χρήστης μπορεί να πληκτρολογήσει τον επιθυμητό κόμβο στο σχετικό πεδίο της κάρτας (το πεδίο Selected Object στην εικόνα) ή να τον επιλέξει από τη λίστα[2].



Σχήμα 4.4: Επιλογή λειτουργίας και κόμβου για ενημέρωση

Η διεργασία **Describe Class** είναι παραμετρική. Δίνοντας σαν παράμετρο το *Artificial Object* (στην κάρτα επιλογής διεργασίας), παίρνουμε για πεδίο δράσης της διεργασίας ένα σύνολο κλάσεων, που στην προκειμένη περίπτωση είναι οι υποκλάσεις του *Artificial Object*. Τμήμα της λίστας βλέπουμε στο σχήμα (List of Classes). Προσφέρουμε λειτουργίες δημιουργίας, διαγραφής, μετονομασίας, ταξινόμησης, γενίκευσης, ενημέρωσης γνωρισμάτων. Μάλιστα έχουμε ορίσει λειτουργίες, όπως οι **Edit Attributes** και **Edit Comment**, που είναι του ίδιου τύπου (UPDATE_ATTRIBUTES) αλλά επειδή έχουν διαφορετικό σύνολο ορισμάτων καθορισμένο στο μοντέλο διεργασιών, διαφοροποιούνται και προσφέρονται σαν ξεχωριστές επιλογές στο χρήστη.

Στην ειδική περίπτωση που το πεδίο δράσης δεν καθορίζεται από το μοντέλο, δεν παρουσιάζουμε όλες τις οντότητες της βάσης σε λίστα αφού κάτι τέτοιο θα ήταν αναποτελεσματικό. Οπως έχουμε εξηγήσει, σε αυτή την περίπτωση το πεδίο δράσης της διεργασίας είναι όλοι οι κόμβοι της βάσης. Περιμένουμε από το χρήστη να πληκτρολογήσει τον επιθυμητό κόμβο στο σχετικό πεδίο.

Μετά την επιλογή του κόμβου, πρέπει να επιλέξουμε τη λειτουργία που θα εφαρμόσουμε σε αυτόν. Ενα σύνολο επιλογών στα αριστερά της κάρτας μάς δίνει τις προσφερόμενες λειτουργίες. Αυτές είναι τόσες, όσες και οι λειτουργίες που περιγράφονται στο μοντέλο για την τρέχουσα διεργασία. Η λειτουργία που διαλέγουμε ενεργεί στη προεπιλεγμένη οντότητα (εκτός αν είναι τύπου CREATE_NODE οπότε η οντότητα αγνοείται). Ανάλογα με τον τύπο της λειτουργίας που επιλέγουμε,

[2]Εξαίρεση σε αυτή τη διαδικασία αποτελεί η περίπτωση που θέλουμε να δημιουργήσουμε έναν κόμβο, με λειτουργία τύπου CREATE_NODE. Τότε δε χρειάζεται να επιλέξουμε κόμβο από το πεδίο δράσης της διεργασίας, αφού πρόκειται να δημιουργήσουμε έναν καινούργιο.

παρουσιάζεται η κατάλληλη κάρτα ενημέρωσης.

4.4.2 Καθορισμός λειτουργίας για γνωρίσματα

Τα γνωρίσματα όπως έχουμε ήδη αναφέρει είναι εξαρτημένα από την οντότητα εκκίνησής τους. Για να προσδιορίσουμε ένα γνώρισμα πρέπει να αναφερθούμε στην οντότητα εκκίνησης και το όνομά του. Αν δεν έχει όνομα τότε εκτός από την οντότητα εκκίνησης θα χρειαστούμε ακόμα τις κατηγορίες στις οποίες ανήκει και την τιμή του. Σε κάθε περίπτωση όμως χρειαζόμαστε την οντότητα εκκίνησης που μπορεί με τη σειρά της να είναι γνώρισμα ή κόμβος. Αν είναι κόμβος μπορούμε να τον προσδιορίσουμε από το πεδίο δράσης κάποιας διεργασίας, όπως είδαμε στην ενότητα 4.4.1. Αν όμως είναι γνώρισμα πρέπει να επαναλάβουμε την παραπάνω διαδικασία για τον προσδιορισμό του.

Είναι λογικό λοιπόν να ξεκινάμε τη διαδικασία προσδιορισμού ενός γνωρίσματος από μια κάρτα που να παρουσιάζει τα γνωρίσματα του κόμβου από τον οποίο τελικά[3] εκκινεί το γνώρισμα. Η κάρτα αυτή είναι η κάρτα ενημέρωσης γνωρισμάτων, που φαίνεται στο σχήμα 4.5, ενώ παρουσιάζεται αναλυτικά στην ενότητα 4.5.6. Δίνουμε στο χρήστη τη δυνατότητα, μέσω αυτής της κάρτας, να δημιουργεί και να διαγράφει γνωρίσματα της εστιακής οντότητας που βρίσκονται στις ενημερώσιμες κατηγορίες της κάρτας, αλλά συγχρόνως να μπορεί να εκτελεί ενημερώσεις μετονομασίας, ταξινόμησης, γενίκευσης και απόδοσης γνωρίσματος για τα γνωρίσματα που η κάρτα προσφέρει προς ενημέρωση. Οι δύο πρώτες ενημερώσεις προσφέρονται σε δύο ζεύγη κουμπιών δύο καταστάσεων (<existing> - <to be deleted>) και (<to be added> - <cancel addition>). Οι τέσσερις επόμενες ενημερώσεις προσφέρονται με τη μορφή μενού επιλογών, όπως φαίνεται στο σχήμα 4.5. Η πιο πρόσφατα επιλεγμένη λειτουργία του μενού, αποτελεί κατάσταση για την κάρτα ενημέρωσης γνωρισμάτων, με την έννοια ότι σε κάθε γνώρισμα που επιλέγει[4] ο χρήστης, εφαρμόζεται η ενημέρωση που αποτελεί τρέχουσα κατάσταση για την κάρτα.

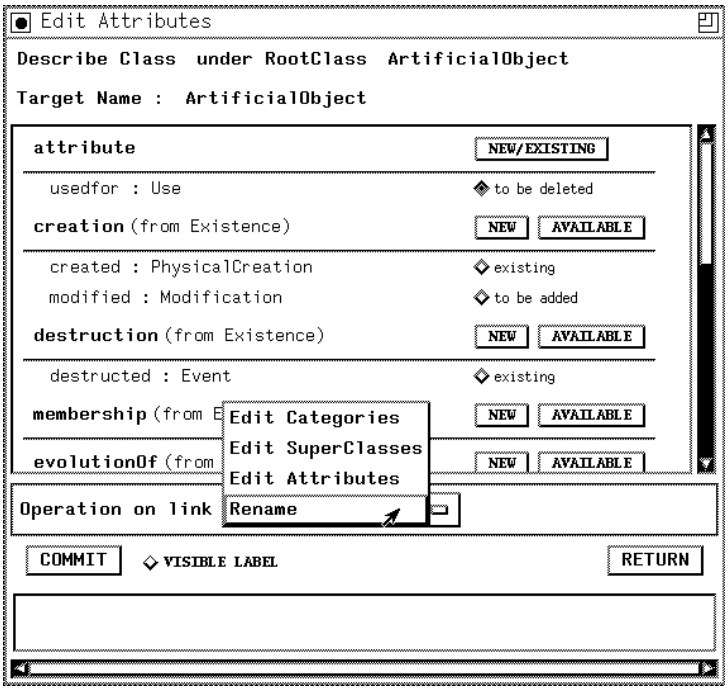
Για κάθε γνώρισμα που εκκινεί από άλλο γνώρισμα, μπορούμε από αυτή την κάρτα να ζητήσουμε κάρτες ενημέρωσης γνωρισμάτων για γνωρίσματα, μέχρι να βρούμε το επιθυμητό (δες σχήμα 4.6).

4.5 Κάρτες ενημέρωσης

Εχουμε δημιουργήσει ένα σύνολο από κάρτες ενημέρωσης με τις οποίες μπορούμε να εκτελέσουμε όλες τις στοιχειώδεις ενημερώσεις που αντιστοιχούν σε κάθε τύπο λειτουργίας ενημέρωσης. Κάθε κάρτα αντιστοιχεί σε μία λειτουργία ενημέρωσης, ενώ έχει μια οντότητα σαν εστιακή. Θα την καλούμε εστιακή οντότητα (ΕΟ), κατ' αναλογία με την ορολογία που χρησιμοποιούμε στις λειτουργίες ενημέρωσης. Ολες οι πράξεις ενημέρωσης που γίνονται στην κάρτα, αφορούν αυτή την οντότητα.

[3] Το γνώρισμα, ως εξαρτημένη οντότητα, έχει σαν οντότητα εκκίνησης ένα κόμβο ή ένα γνώρισμα που εκκινεί από κόμβο κοκ.

[4] Το γνώρισμα επιλέγεται αν πατήσουμε το αριστερό πλήκτρο του ποντικιού πάνω σε αυτό.



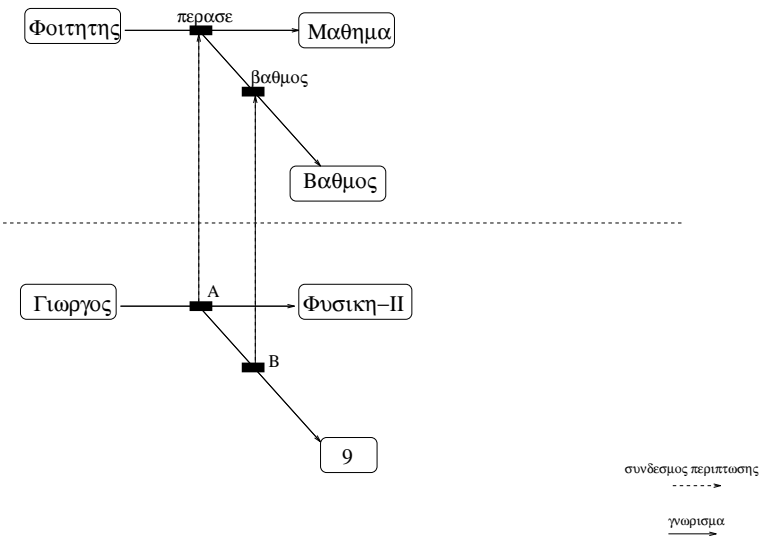
Σχήμα 4.5: Κάρτα ενημέρωσης γνωρισμάτων

Η εστιακή οντότητα **ArtificialObject** έχει ενημερώσιμες κατηγορίες τις *attribute*, *creation*, *destruction* κοκ. Ετσι βλέπουμε πως για την κατηγορία (**Existence**, *destruction*) υπάρχει το γνώρισμα *destructed*, με τιμή γνωρίσματος το **Event** (ένδειξη <existing>). Με την κάρτα ενημέρωσης γνωρισμάτων δίνουμε στο χρήστη τη δυνατότητα να δημιουργεί και να διαγράφει γνωρίσματα (ενδείξεις <to be added>, <to be deleted>). Επίσης με ένα μενού επιλογών προσφέρονται ενημερώσεις μετονομασίας, ταξινόμησης, γενίκευσης και απόδοσης γνωρίσματος για τα γνωρίσματα που η κάρτα παρουσιάζει. Η πιο πρόσφατα επιλεγμένη ενημέρωση του μενού αποτελεί κατάσταση για την κάρτα, με την έννοια ότι αυτή είναι η ενημέρωση που δρα σε κάθε γνώρισμα που επιλέγουμε στη συνέχεια για ενημέρωση.

Ολες οι κάρτες έχουν το χαρακτηριστικό ότι διαθέτουν ένα κουμπί COMMIT με το οποίο ο χρήστης ζητά να γίνουν δεκτές οι αλλαγές που έκανε στη συγκεκριμένη κάρτα, για την ΕΟ. Οι αλλαγές της κάρτας, όσες κι αν είναι, μαζεύονται και δίνονται στη βάση σε μια δοσοληψία, έτσι ώστε είτε όλες να γίνουν δεκτές είτε καμία. Με αυτό τον τρόπο διατηρείται η ακεραιότητα της βάσης.

Από τις κάρτες ενημέρωσης που έχουμε κατασκευάσει, άλλες δέχονται σαν ΕΟ κόμβους κι άλλες γνωρίσματα. Δύο από αυτές (δημιουργίας κόμβου και διαγραφής κόμβου) έχουν ΕΟ αποκλειστικά κόμβους. Τέσσερις άλλες (μετονομασίας, ταξινόμησης, γενίκευσης, ενημέρωσης γνωρισμάτων) μπορούν να συναντηθούν σε δύο μορφές έχοντας ΕΟ κόμβο είτε γνώρισμα. Ειδικά για την κάρτα ενημέρωσης γνωρισμάτων έχουμε μια ακόμα εξειδικευμένη χρήση, αυτή της ενημέρωσης σύνθετων οντοτήτων (δες ενότητα 6.1.2).

Ολες οι κάρτες ενημέρωσης με ΕΟ κόμβο, καλούνται από την κάρτα επιλογής λειτουργίας κόμβου, όπως αυτή που είδαμε στο σχήμα 4.4. Μια από τις κάρτες ενημέρωσης, η κάρτα ενημέρωσης γνωρισμάτων, είναι υπεύθυνη για την κλήση καρτών με ΕΟ γνώρισμα. Για αυτό το λόγο, όταν στη συνέχεια εμφανίζονται στα σχήματα κάρτες με ΕΟ γνώρισμα, εμφανίζεται μαζί και η κάρτα ενημέρωσης γνωρισμάτων της οντότητας από την οποία το γνώρισμα εκκινεί. Σε αυτές



Σχήμα 4.6: Ενημέρωση γνωρίσματος που εκκινεί από γνώρισμα

Για να ενημερώσουμε το γνώρισμα *B* πρέπει πρώτα να το εντοπίσουμε: Από την κάρτα ενημέρωσης γνωρισμάτων του κόμβου **Γιώργος**, βρίσκουμε το γνώρισμα *A* που τον συνδέει με το Μάθημα **Φυσική-ΙΙ**, μέσω της κατηγορίας (**Φοιτητής**, *πέρασε*). Ζητούμε την κάρτα ενημέρωσης γνωρισμάτων για το *A* (ενημέρωση *Edit Attributes* στο μενού επιλογών ενημερώσεων για γνωρίσματα). Βρίσκουμε σε αυτή την κάρτα το *B* και τότε μπορούμε να το ενημερώσουμε με κάποια από τις λειτουργίες που προσφέρει η κάρτα ενημέρωσης γνωρισμάτων. Αν το *B* δεν υπάρχει μπορούμε από την κάρτα ενημέρωσης γνωρισμάτων του *A* να το δημιουργήσουμε, σαν περίπτωση της κατηγορίας ((**Φοιτητής**, *πέρασε*), *βαθμός*).

τις περιπτώσεις, που περισσότερες από μια κάρτες ενημέρωσης εμφανίζονται σε μια ακολουθία, τη δικαιοδοσία για ενημέρωση της βάσης με τις αλλαγές που όλες μαζί επιβάλλουν, έχει η τελευταία κάρτα της ακολουθίας.

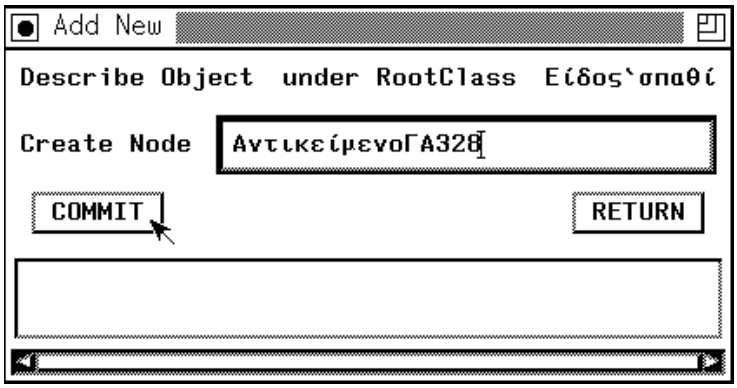
4.5.1 Δημιουργία κόμβου

Η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου CREATE_NODE. Σε αυτήν την κάρτα ο χρήστης καλείται να καθορίσει τον κόμβο που θα δημιουργηθεί, δίνοντας το λογικό του όνομα. Αυτό αρκεί για να προσδιοριστεί μοναδικά ένας κόμβος.

Οι κλάσεις στις οποίες αυτός πρόκειται να ταξινομηθεί δίνονται από το σύνολο ορισμάτων της λειτουργίας, πληροφορία που βρίσκεται στο μοντέλο διεργασιών. Αν η συγκεκριμένη λειτουργία δεν έχει σύνολο ορισμάτων συνδεδεμένο με αυτήν, ένα ακόμα βοηθητικό παράθυρο εμφανίζεται και ο χρήστης καλείται να δώσει τον κόμβο στον οποίο η οντότητα θα γίνει περίπτωση. Μπορεί εναλλακτικά, αντί για κλάση καθορισμένη από χρήστη, να δώσει το επίπεδο ταξινόμησης στο οποίο θέλει να την κατατάξει. Παράδειγμα κάρτας δημιουργίας κόμβου βλέπουμε στο σχήμα 4.7.

4.5.2 Διαγραφή κόμβου

Η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου DELETE_NODE. Αν το σύνολο ορισμάτων της λειτουργίας είναι ορισμένο στο μοντέλο διεργασιών, τότε αυτό καθορίζει ποιες είναι οι οντότητες

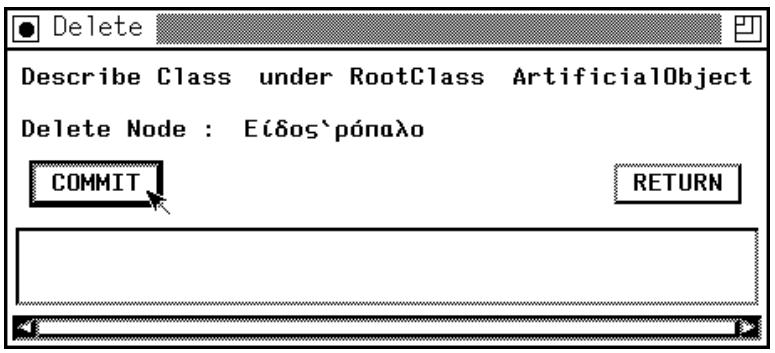


Σχήμα 4.7: Κάρτα δημιουργίας κόμβου

Ο κόμβος **ΑντικείμενοΓΑ328** που πρόκειται να δημιουργηθεί μετά την επιλογή COMMIT, θα γίνει περίπτωση της κλάσης **Είδος'σπαθί**, γεγονός που καθορίζεται από το σύνολο ορισμάτων της λειτουργίας.

που επιτρέπεται να διαγραφούν. Αν δεν υπάρχει, τότε χωρίς περιορισμούς από το μοντέλο, η εστιακή οντότητα μπορεί να διαγραφεί.

Σε αυτήν την κάρτα ο χρήστης καλείται απλώς να επιβεβαιώσει ή να ακυρώσει τη διαγραφή της εστιακής οντότητας. Η διαγραφή της οντότητας μπορεί να οδηγήσει και σε άλλες πράξεις που θα γίνουν για να διατηρηθούν περιορισμοί ακεραιότητας που τυχόν έχουν οριστεί στο μοντέλο περιορισμών της βάσης (δες ενότητες 5.1.2, 5.4). Παράδειγμα κάρτας διαγραφής κόμβου βλέπουμε στο σχήμα 4.8.



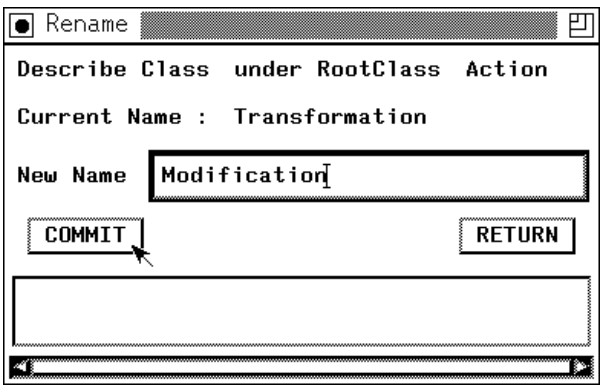
Σχήμα 4.8: Κάρτα διαγραφής κόμβου

Με την επιλογή COMMIT γίνεται η διαγραφή του κόμβου **Είδος'ρόπαλο** στη βάση.

4.5.3 Μετονομασία οντότητας

Η κάρτα αυτή μπορεί να εμφανιστεί σε δύο μορφές: έχοντας σαν ΕΟ ένα κόμβο ή ένα γνώρισμα. Σε αυτήν την κάρτα, ο χρήστης καλείται να δώσει στο κατάλληλο πεδίο, το όνομα το οποίο θα αντικαταστήσει το λογικό όνομα της ΕΟ.

Μετονομασία κόμβου Στην περίπτωση που η ΕΟ είναι κόμβος, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου RENAME_NODE. Αν το σύνολο ορισμάτων της λειτουργίας είναι ορισμένο στο μοντέλο διεργασιών, τότε αυτό καθορίζει ποιες είναι οι οντότητες που επιτρέπεται να μετονομαστούν. Αν δεν υπάρχει τότε, χωρίς περιορισμούς από το μοντέλο, η εστιακή οντότητα μπορεί να μετονομαστεί. Παράδειγμα κάρτας μετονομασίας κόμβου βλέπουμε στο σχήμα 4.9.



Σχήμα 4.9: Κάρτα μετονομασίας κόμβου

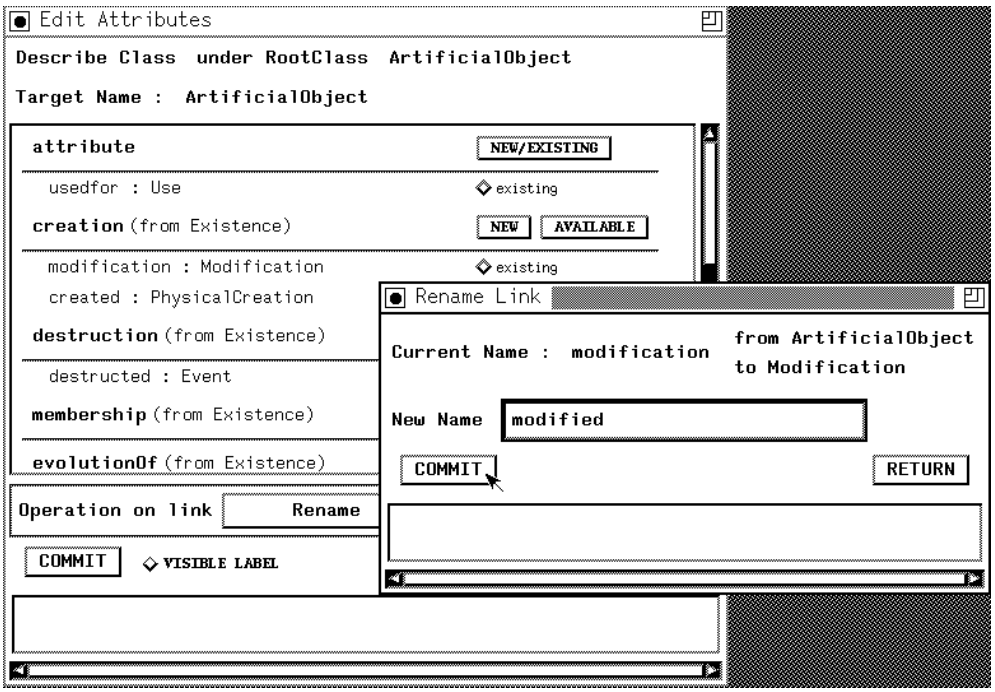
Γράφουμε το καινούργιο λογικό όνομα **Modification** στο κατάλληλο πεδίο (New Name) και πατώντας το COMMIT ο κόμβος με όνομα **Transformation** μετονομάζεται.

Μετονομασία γνωρίσματος Στην περίπτωση που η ΕΟ είναι γνώρισμα, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου UPDATE_ATTRIBUTES και συγκεκριμένα στην ενημέρωση μετονομασία γνωρίσματος. Παράδειγμα κάρτας μετονομασίας γνωρίσματος βλέπουμε στο σχήμα 4.10.

4.5.4 Ταξινόμηση οντότητας

Η κάρτα αυτή μπορεί να εμφανιστεί σε δύο μορφές: έχοντας σαν ΕΟ ένα κόμβο ή ένα γνώρισμα. Σε αυτήν παρουσιάζονται οι οντότητες των οποίων η εστιακή οντότητα είναι περίπτωση. Οσες οντότητες επιτρέπεται να διαγραφούν από κλάσεις της ΕΟ, εμφανίζονται στην κατάσταση <existing>. Αυτές με επιλογή του χρήστη μπορούν να διαγραφούν από κλάσεις της ΕΟ, κάνοντας την κατάστασή τους <to be deleted>. Ακύρωση της προσπάθειας για διαγραφή τους, επιστρέφει στην κατάσταση <existing>. Ετσι βλέπουμε πως αυτές οι δύο καταστάσεις είναι συμπληρωματικές και μπορούμε να μεταβούμε από τη μια στην άλλη. Οσες δεν επιτρέπεται να διαγραφούν εμφανίζονται στην κατάσταση <readonly>.

Κάθε φορά που προσθέτουμε μια μελλοντική κλάση της εστιακής οντότητας, αυτή εμφανίζεται στην κατάσταση <to be added>, ενώ ακύρωση της προσπάθειας για εισαγωγή της, οδηγεί στην κατάσταση <cancel addition>. Οι ενδείξεις <to be added> και <cancel addition> είναι συμπληρωματικές και μπορούμε να μεταβούμε από τη μια στην άλλη.



Σχήμα 4.10: Κάρτα μετονομασίας γνωρίσματος

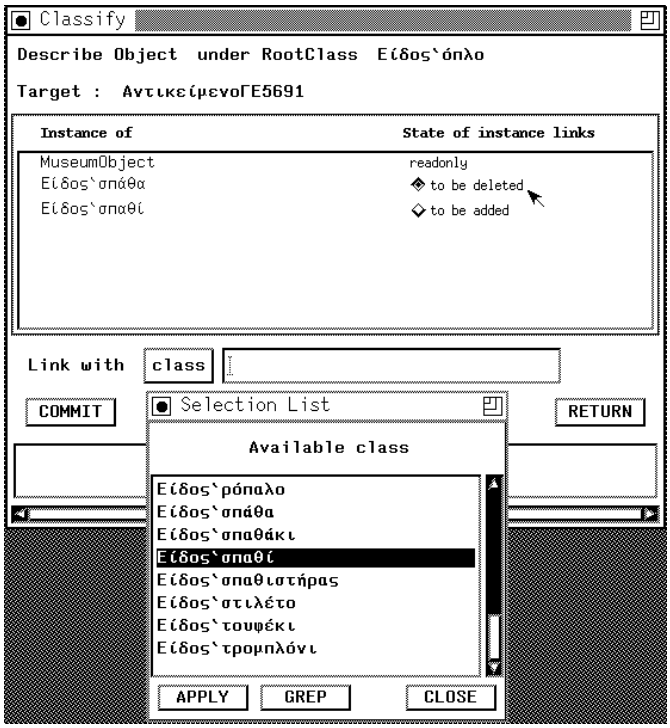
Βρίσκουμε το γνώρισμα *modification* από την κάρτα ενημέρωσης γνωρισμάτων του **ArtificialObject** και επιλέγουμε λειτουργία μετονομασίας γνωρίσματος. Γράφουμε το καινούργιο λογικό όνομα *modified* στο κατάλληλο πεδίο, και πατώντας το COMMIT το γνώρισμα μετονομάζεται.

Ταξινόμηση κόμβου Στην περίπτωση που η ΕΟ είναι κόμβος, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου EDIT_CLASSES. Η λειτουργία μπορεί, προαιρετικά, να έχει σύνολο ορισμάτων στο μοντέλο διεργασιών.

- Αν η λειτουργία έχει σύνολο ορισμάτων τότε μόνο όσες από τις κλάσεις της ΕΟ ανήκουν σε αυτό επιτρέπεται να διαγραφούν (άρα εμφανίζονται στην κατάσταση <existing>). Οσες από τις κλάσεις της ΕΟ δεν ανήκουν στο σύνολο ορισμάτων της λειτουργίας, δεν επιτρέπεται να διαγραφούν και εμφανίζονται στην κατάσταση <readonly>. Επίσης μπορούμε να προσθέσουμε μόνο κλάσεις που ανήκουν στο σύνολο ορισμάτων της λειτουργίας, είτε γράφοντάς τις στο κατάλληλο πεδίο της κάρτας είτε, εναλλακτικά, επιλέγοντάς τις από μια λίστα. Τα περιεχόμενα της λίστας είναι οι οντότητες αυτού του συνόλου.

- Αν η λειτουργία δεν έχει σύνολο ορισμάτων, μπορούμε να διαγράψουμε οποιαδήποτε από τις κλάσεις της ΕΟ. Επίσης μπορούμε να προσθέσουμε οποιαδήποτε κλάση σε αυτό, την οποία όμως πρέπει να πληκτρολογήσουμε στο κατάλληλο πεδίο. Δεν έχουμε σε αυτή την περίπτωση τη δυνατότητα για προτεινόμενες κλάσεις.

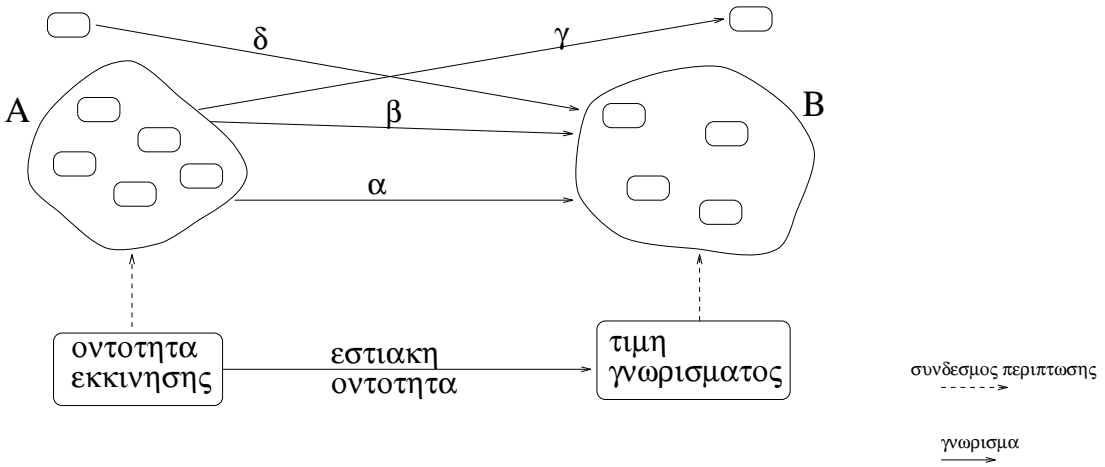
Παράδειγμα κάρτας ταξινόμησης για κόμβους έχουμε στο σχήμα 4.11.



Σχήμα 4.11: Κάρτα ταξινόμησης κόμβου

Το εστιακό αντικείμενο **ΑντικείμενοΓΕ5691** έχει ταξινομηθεί αρχικά στην κλάση **Είδος'σπάθα**, από την οποία και ζητούμε τη διαγραφή του (κατάσταση <to be deleted>). Ζητούμε την ταξινόμησή του στην κλάση **Είδος'σπαθί** (κατάσταση <to be added>). Το αντικείμενο είναι ακόμα ταξινομημένο στην κλάση **MuseumObject** από την οποία η συγκεκριμένη λειτουργία δεν επιτρέπει τη διαγραφή (κατάσταση <readonly>). Αυτό είναι αποτέλεσμα του γεγονότος ότι το σύνολο ορισμάτων της λειτουργίας δεν περιέχει την εν λόγω κλάση, για λόγους ασφάλειας. Οι κλάσεις στις οποίες μπορούμε να ταξινομήσουμε την εστιακή οντότητα φαίνονται στην παρουσιαζόμενη λίστα. Αυτές περιέχονται στο σύνολο ορισμάτων της λειτουργίας, στην οποία η κάρτα αντιστοιχεί

Ταξινόμηση γνωρίσματος Στην περίπτωση που η ΕΟ είναι γνώρισμα, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου UPDATE_ATTRIBUTES και συγκεκριμένα στην ενημέρωση ταξινόμηση γνωρίσματος. Ολοι οι σύνδεσμοι περίπτωσης της ΕΟ με τις κατηγορίες της επιτρέπεται να διαγραφούν (άρα εμφανίζονται στην κατάσταση <existing>). Για την δημιουργία συνδέσμων περίπτωσης, επειδή είναι δύσκολο να ορίσει ο χρήστης μοναδικά μια κατηγορία (χρειάζεται την οντότητα εκκίνησής της και το λογικό της όνομα), δεν προσφέρουμε τη δυνατότητα να πληκτρολογείται η επιθυμητή κατηγορία σε κάποιο πεδίο. Προσφέρουμε μόνο τη δυνατότητα επιλογής από μια λίστα που περιέχει τις κατηγορίες του γνωρίσματος, στις οποίες επιτρέπεται να ταξινομηθεί, σύμφωνα με τους περιορισμούς που επιβάλλονται από τη γλώσσα Telos. Αυτές υπολογίζονται εύκολα, όπως βλέπουμε στο σχήμα 4.12. Παράδειγμα κάρτας ταξινόμησης για γνωρίσματα έχουμε στο σχήμα 4.13.



Σχήμα 4.12: Εύρεση διαθέσιμων κατηγοριών

Η *εστιακή οντότητα*, όπως φαίνεται στο σχήμα, είναι ένα γνώρισμα. Στο σύνολο **A** περιέχονται οι κλάσεις της οντότητας εκκίνησης. Από αυτές εκκινούν τα γνωρίσματα α, β, γ. Στο σύνολο **B** περιέχονται οι κλάσεις της τιμής γνωρίσματος, στις οποίες καταλήγουν τα γνωρίσματα α, β, δ. Η *εστιακή οντότητα* μπορεί να γίνει περίπτωση μόνο στις κατηγορίες α και β, σύμφωνα με τους περιορισμούς που επιβάλλει η Telos.

4.5.5 Γενίκευση οντότητας

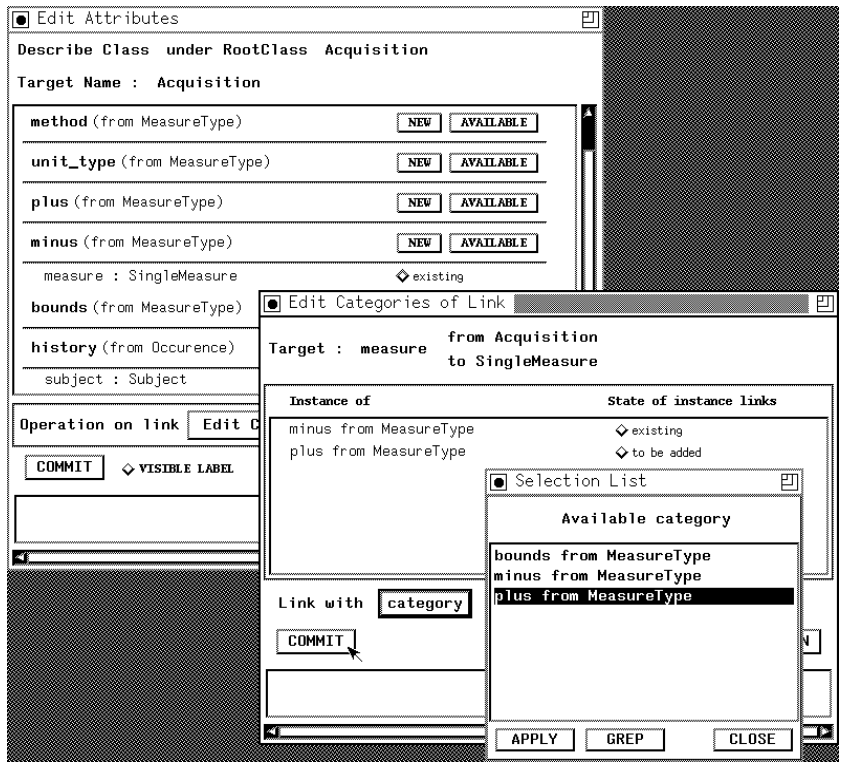
Με τους ίδιους μηχανισμούς της κάρτας ταξινόμησης, λειτουργεί και η κάρτα γενίκευσης. Και εδώ η ΕΟ μπορεί να είναι κόμβος είτε γνώρισμα.

Γενίκευση κόμβου Στην περίπτωση που η ΕΟ είναι κόμβος, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου EDIT_SUPERCLASSES. Η κάρτα αυτή έχει την ίδια λειτουργικότητα με την κάρτα ταξινόμησης κόμβου, μόνο που ενημερώνει συνδέσμους εξειδίκευσης, αντί συνδέσμους περίπτωσης. Παράδειγμα χρήσης της βλέπουμε στο σχήμα 4.14.

Γενίκευση γνωρίσματος Στην περίπτωση που η ΕΟ είναι γνώρισμα, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου UPDATE_ATTRIBUTES και συγκεκριμένα στην ενημέρωση γενίκευση γνωρίσματος. Η κάρτα αυτή έχει την ίδια λειτουργικότητα με την κάρτα ταξινόμησης γνωρίσματος, μόνο που ενημερώνει συνδέσμους εξειδίκευσης, αντί συνδέσμους περίπτωσης. Παράδειγμα χρήσης της βλέπουμε στο σχήμα 4.15

4.5.6 Ενημέρωση γνωρισμάτων

Η κάρτα ενημέρωσης γνωρισμάτων είναι η πιο σύνθετη από τις κάρτες ενημέρωσης. Ο χρήστης μπορεί από αυτήν να δημιουργήσει ή να διαγράψει γνωρίσματα που εκκινούν από την ΕΟ, μπορεί να ζητήσει ενημερώσεις μετονομασίας, ταξινόμησης, γενίκευσης και απόδοσης γνωρίσματος για αυτά (όπως είδαμε στην ενότητα 4.4.2), ενώ τέλος μπορεί να ζητήσει ενημέρωση γνωρισμάτων σύνθετης



Σχήμα 4.13: Κάρτα ταξινόμησης γνωρίσματος

Βρίσκουμε το γνώρισμα *measure* στην κάρτα ενημέρωσης γνωρισμάτων του κόμβου **Acquisition** και ζητούμε ταξινόμησή του. Το γνώρισμα έχει αρχικά ταξινομηθεί στην κατηγορία (**MeasureType**, *minus*). Μετά το COMMIT, το γνώρισμα γίνεται περίπτωση και της κατηγορίας (**MeasureType**, *plus*). Στη βοηθητική κάρτα με τις διαθέσιμες κατηγορίες (Available category), βλέπουμε ποιες είναι οι κατηγορίες στις οποίες το γνώρισμα επιτρέπεται να ταξινομηθεί (στηριζόμενοι στους δομικούς περιορισμούς που επιβάλλει η Telos).

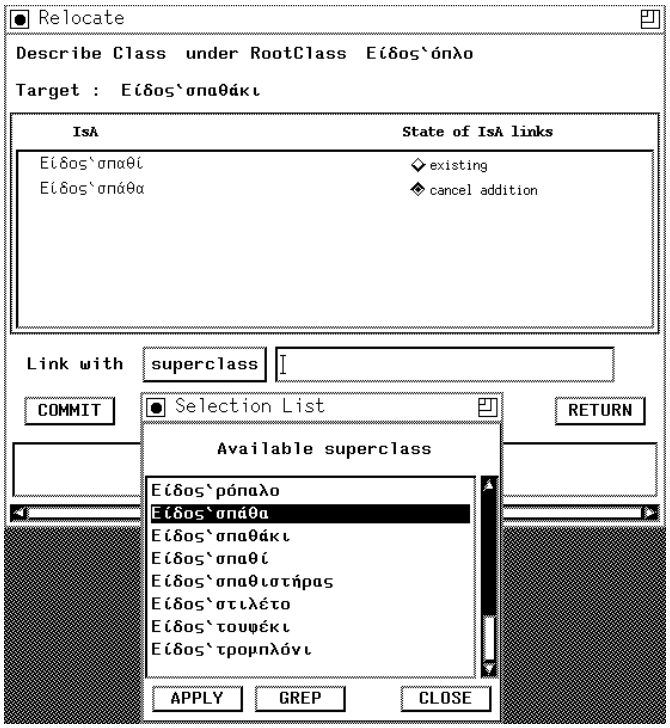
οντότητας, όπως θα δούμε στην ενότητα 6.1.2. Εδώ θα κάνουμε μια γενική περιγραφή της κάρτας και θα περιοριστούμε στη λειτουργικότητά της, όσον αφορά στη δημιουργία/διαγραφή γνωρισμάτων.

Η κάρτα αυτή μπορεί να εμφανιστεί σε δύο μορφές: έχοντας σαν ΕΟ ένα κόμβο ή ένα γνώρισμα. Σκοπός της είναι, ανάμεσα στα άλλα, να εμφανίζει ενημερώσιμες κατηγορίες και να δημιουργεί/διαγράφει γνωρίσματα, που εκκινούν από την ΕΟ , τα οποία όμως είναι στην εμβέλεια αυτών των κατηγοριών.

Παρόλο που τα γνωρίσματα είναι οντότητες, έχουμε δύο βασικούς λόγους που χειριζόμαστε τη δημιουργία/διαγραφή τους από μία κάρτα, σε αντίθεση με τις ξεχωριστές κάρτες δημιουργίας και διαγραφής κόμβου:

1. Τα γνωρίσματα δεν μπορούν να προσδιοριστούν χωρίς την οντότητα από την οποία εκκινούν. Σε αυτήν την οντότητα αποδίδουμε γνωρίσματα και από αυτήν τα διαγράφουμε, επομένως αυτή πρέπει να είναι η εστιακή οντότητα της κάρτας.

2. Θέλουμε να δώσουμε στο χρήστη την αίσθηση ότι το γνώρισμα είναι ένας σύνδεσμος, όπως οι σύνδεσμοι περίπτωσης και εξειδίκευσης. Αυτή είναι και η διαισθητική αντίληψη που έχει



Σχήμα 4.14: Κάρτα γενίκευσης κόμβου

Η εστιακή οντότητα **Είδος'σπαθάκι** είναι εξειδίκευση της κλάσης **Είδος'σπαθί** (κατάσταση <existing>). Ατυχώς την κατατάσσουμε στην κλάση **Είδος'σπάθα** οπότε και αναιρούμε αυτή την πράξη (κατάσταση <cancel addition>).

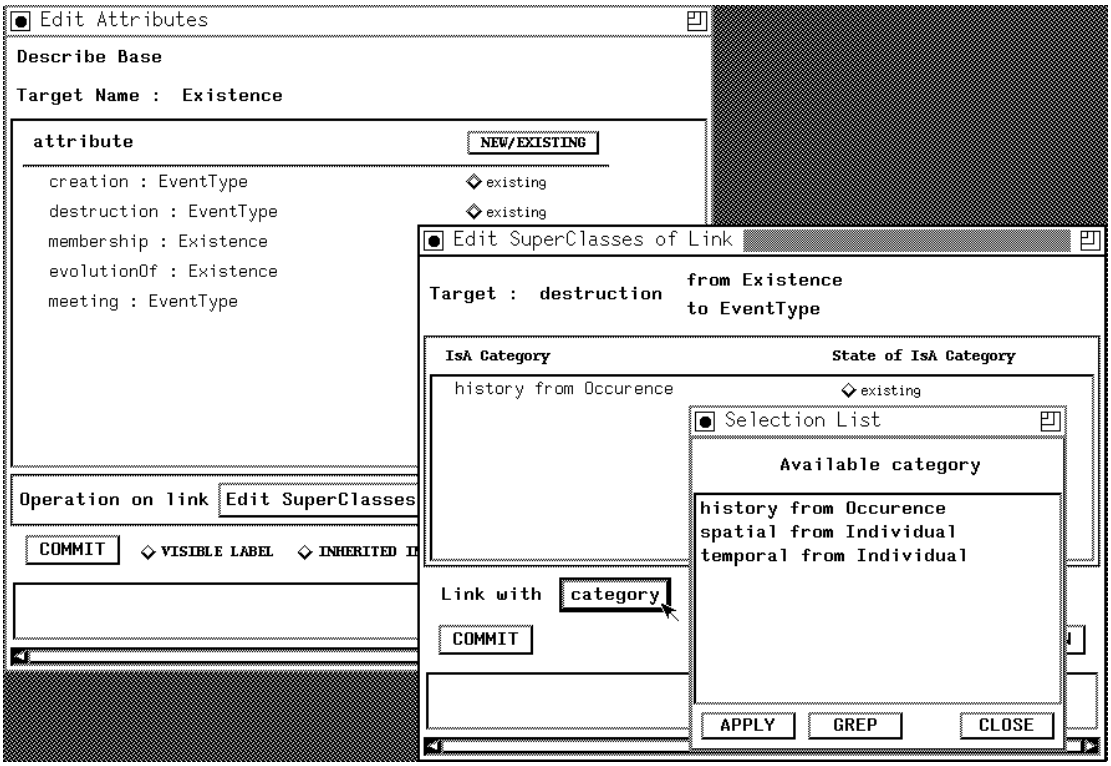
κανείς για τα γνωρίσματα. Η διαφορά είναι μόνο ότι λόγω της ιδιαιτερότητας της υλοποιησής τους από την Telos, μπορούμε να αποδώσουμε στους συνδέσμους αυτούς ονόματα, να τους ταξινομούμε και να τους εξειδικεύουμε σε κατηγορίες, ενώ τέλος μπορούμε να αποδίδουμε σε αυτούς γνωρίσματα.

Για τους παραπάνω λόγους, η κάρτα αυτή μοιάζει στις βασικές λειτουργίες της με τις κάρτες ταξινόμησης και γενίκευσης, παρέχοντας εποπτική εικόνα των συνδέσμων που δημιουργούνται ή διαγράφονται από την εστιακή οντότητα.

Ενημέρωση γνωρισμάτων κόμβου Στην περίπτωση που η ΕΟ είναι κόμβος, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου UPDATE_ATTRIBUTES. Η λειτουργία μπορεί, προαιρετικά, να έχει σύνολο ορισμάτων στο μοντέλο διεργασιών.

- Αν η λειτουργία έχει σύνολο ορισμάτων, τότε οι κατηγορίες που αυτό περιέχει είναι ενημερώσιμες. Αυτό σημαίνει ότι μόνο τα γνωρίσματα της ΕΟ που ανήκουν σε αυτές τις κατηγορίες, μπορούν να ενημερωθούν.
- Αν η λειτουργία δεν έχει σύνολο ορισμάτων τότε όλες οι κατηγορίες, που βρίσκονται με τη βοήθεια των κλάσεων της ΕΟ, είναι ενημερώσιμες.

Παράδειγμα κάρτας ενημέρωσης γνωρισμάτων κόμβου είδαμε στο σχήμα 4.5.



Σχήμα 4.15: Κάρτα γενίκευσης γνωρίσματος

Βρίσκουμε το γνώρισμα *destruction* στην κάρτα ενημέρωσης γνωρισμάτων του κόμβου **Existence** και ζητούμε γενίκευσή του. Βλέπουμε ότι έχει υπερκλάση την κατηγορία (**Occurence**, *history*). Με το κουμπί *category*, βλέπουμε σε ποιες άλλες κατηγορίες επιτρέπεται να γενικεύσουμε, την εστιακή οντότητα (στηριζόμενοι στους δομικούς περιορισμούς που επιβάλλει η Telos).

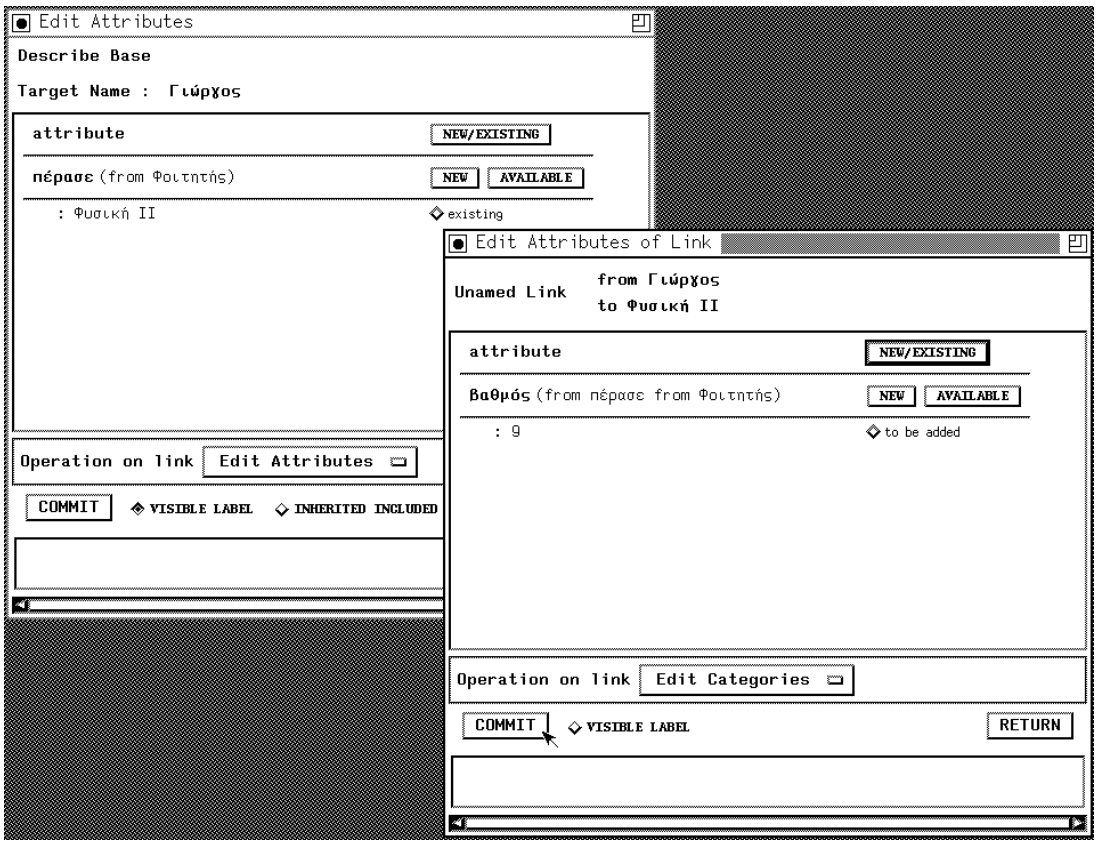
Ενημέρωση γνωρισμάτων γνωρίσματος Στην περίπτωση που η ΕΟ είναι γνώρισμα, η κάρτα αντιστοιχεί σε λειτουργίες ενημέρωσης τύπου UPDATE_ATTRIBUTES και συγκεκριμένα στην ενημέρωση απόδοσης γνωρίσματος. Παράδειγμα κάρτας ενημέρωσης γνωρισμάτων γνωρίσματος, για την υποθετική περίπτωση που δείξαμε στο σχήμα 4.6, βλέπουμε στο σχήμα 4.16.

4.5.6.1 Καταστάσεις της κάρτας

Η κάρτα έχει δύο μεταβλητές κατάστασης, οι οποίες ρυθμίζουν το είδος και το πλήθος της πληροφορίας που εμφανίζεται σε αυτήν. Ο λόγος ύπαρξής τους είναι η οικονομία χώρου που επιτυγχάνουμε σε πολλές περιπτώσεις. Η εναλλαγή των καταστάσεων αυτών γίνεται κατ' επιλογή του χρήστη. Οι καταστάσεις στις οποίες αναφερόμαστε είναι:

- Κατάσταση *ορατότητας ονόματος γνωρισμάτων*: Ο χρήστης επιλέγει αν θέλει να είναι εμφανίσιμα τα ονόματα των γνωρισμάτων ή να μένουν κρυφά. Παίρνει τις τιμές HIDDEN LABEL / VISIBLE LABEL.

  Τιμή HIDDEN LABEL δηλώνει μη εμφανίσιμα ονόματα γνωρισμάτων. Είναι χρήσιμη επιλογή



Σχήμα 4.16: Κάρτα ενημέρωσης γνωρισμάτων γνωρίσματος

Στο γνώρισμα (χωρίς όνομα) που εκκινεί από τον κόμβο **Γιώργος** και έχει τιμή γνωρίσματος τον κόμβο **Φυσική ΙΙ**, αποδίδουμε γνώρισμα κατηγορίας ((**Φοιτητής**, *πέρασε*), *βαθμός*), με τιμή γνωρίσματος το βαθμό 9.

όταν περιγράφουμε ατομικά[5] γνωρίσματα, όπου τα ονόματά τους μας είναι συνήθως αδιάφορα. Τιμή VISIBLE LABEL δηλώνει φανερά ονόματα γνωρισμάτων. Είναι χρήσιμη επιλογή όταν καθορίζουμε γνωρίσματα που το λογικό τους όνομα ενδιαφέρει (συνήθως όταν αναφερόμαστε σε γνωρίσματα σε επίπεδο σχήματος).

- Κατάσταση *κληρονομημένων γνωρισμάτων*: Υποδεικνύει τις ενημερώσιμες κατηγορίες των γνωρισμάτων που θα παρουσιαστούν. Παίρνει τις τιμές INHERITED INCLUDED / INHERITED NOT INCLUDED.

  Τιμή INHERITED INCLUDED δηλώνει ότι ο χρήστης επιθυμεί ενημέρωση των γνωρισμάτων στις κατηγορίες που προέρχονται από τις άμεσες αλλά και τις έμμεσες κλάσεις της ΕΟ.

  Τιμή INHERITED NOT INCLUDED δηλώνει ότι ο χρήστης επιθυμεί ενημέρωση των γνωρισμάτων στις κατηγορίες που προέρχονται μόνο από τις άμεσες κλάσεις της εστιακής οντότητας. Είναι χρήσιμη επιλογή όταν θέλουμε περιορισμένο πεδίο κατηγοριών. Ετσι η κάρτα γίνεται πιο ευανάγνωστη ιδίως στις περιπτώσεις μεγάλου πλήθους κατηγοριών.

  Η εναλλαγή της κατάστασης αυτής δεν είναι επιλέξιμη, στην περίπτωση που η κάρτα υλοποιεί

[5] Βρίσκονται στην κατώτερη στάθμη ταξινόμησης.

λειτουργία τύπου UPDATE_ATTRIBUTES με σύνολο ορισμάτων καθορισμένο στο μοντέλο διεργασιών. Οταν αυτό συμβαίνει το σύνολο ορισμάτων της λειτουργίας επιβάλλει ενημέρωση μόνο για συγκεκριμένες κατηγορίες.

4.5.6.2 Παρουσίαση γνωρισμάτων

Κάτω από τις ενημερώσιμες κατηγορίες, βρίσκονται ομαδοποιημένα τα γνωρίσματα της εστιακής οντότητας. Αν ένα γνώρισμα ανήκει σε περισσότερες από μία κατηγορίες, εμφανίζεται σε κάθε μία από αυτές ξεχωριστά. Από τα γνωρίσματα αυτά, άλλα είναι κληρονομημένα από τις υπερκλάσεις της[6], ενώ άλλα εκκινούν από την ίδια.

Τα γνωρίσματα που κληρονομούνται, εμφανίζονται κάτω από την κατηγορία τους, στην κατάσταση <readonly>, ενδεικτικό ότι δεν μπορούν να τροποποιηθούν από αυτή την κάρτα. Είναι γνωρίσματα που τη δικαιοδοσία για τροποποίησή τους, έχουν οι οντότητες από τις οποίες αυτά εκκινούν. Ετσι αν ο χρήστης επιθυμεί να ενημερώσει κάποιο από αυτά, πρέπει να καλέσει την κάρτα ενημέρωσης γνωρισμάτων με ΕΟ εκείνη από την οποία αυτά εκκινούν. Τα γνωρίσματα που εκκινούν από την εστιακή οντότητα της λειτουργίας εμφανίζονται κάτω από την κατηγορία τους, στην κατάσταση <existing>.

4.5.6.3 Δημιουργία γνωρίσματος

Η κάρτα ενημέρωσης γνωρισμάτων μάς δίνει τη δυνατότητα να δημιουργούμε γνωρίσματα που εκκινούν από την εστιακή οντότητα. Ομως εκτός από την οντότητα εκκίνησης τα γνωρίσματα χρειάζονται και μια τιμή προορισμού, την *τιμή γνωρίσματος*, όπως την αποκαλούμε. Προσπάθειά μας είναι να βοηθήσουμε όσο μπορούμε το χρήστη στον καθορισμό του γνωρίσματος. Επομένως είναι επιθυμητό να του προσφέρουμε μηχανισμούς καθοδήγησης για το είδος της τιμής γνωρίσματος που μπορεί να εισαγάγει.

Αν η τιμή γνωρίσματος της κατηγορίας είναι πρωτογενής τύπος, δεν μπορούμε να κάνουμε πρόβλεψη για την τιμή του γνωρίσματος (μπορεί να είναι οποιοδήποτε στοιχείο του πρωτογενούς τύπου). Επιτρέπουμε στο χρήστη να καθορίσει, πληκτρολογώντας, την τιμή γνωρίσματος.

Σε κάθε άλλη περίπτωση ο χρήστης μπορεί να διαλέξει ανάμεσα σε δύο εναλλακτικές λύσεις στο θέμα του καθορισμού τιμής γνωρίσματος[7]:

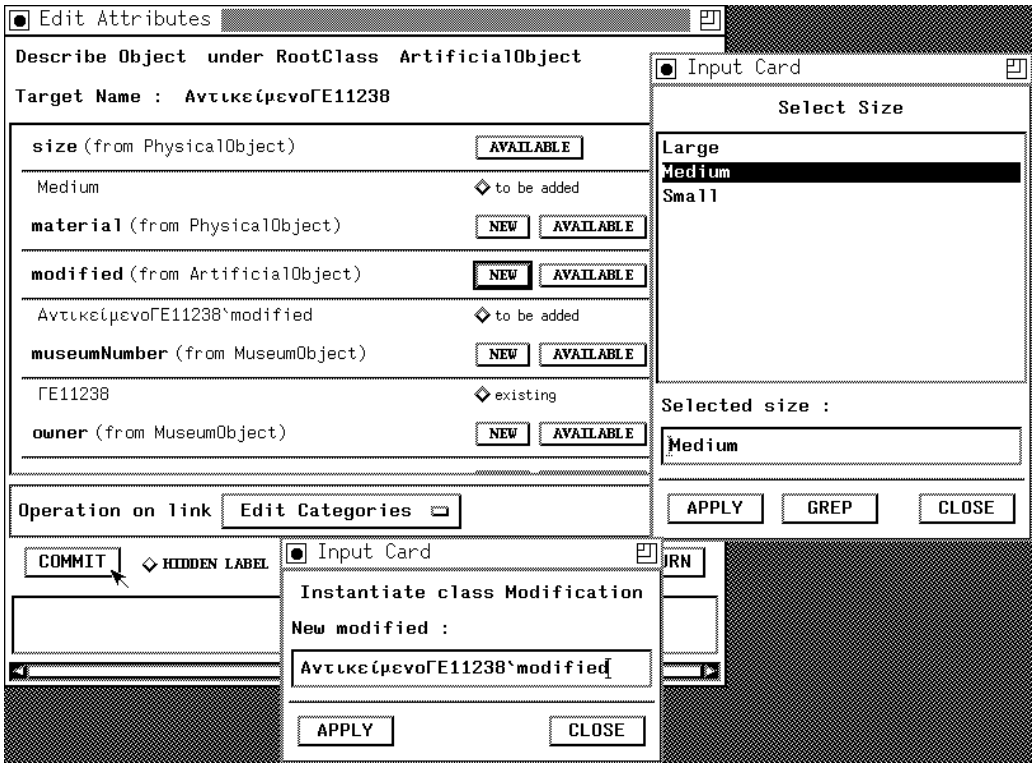
- να συνδέσει την εστιακή οντότητα με ένα υπάρχοντα κόμβο της βάσης
- να δημιουργήσει ένα κόμβο στη βάση και με αυτόν να συνδέσει την εστιακή οντότητα

[6] Σύμφωνα με το μοντέλο δεδομένων της Telos, μια κλάση κληρονομεί τα γνωρίσματα των υπερκλάσεών της.

[7] Υπάρχουν περιπτώσεις όπως θα δούμε στην ενότητα 6.3, όπου κάποια από τις εναλλακτικές λύσεις μπορεί να μην είναι διαθέσιμη, για λόγους διατήρησης εννοιολογικών περιορισμών της βάσης.

Και στις δύο περιπτώσεις πρέπει πρώτα να καθορίσουμε το πεδίο τιμών γνωρίσματος (δες ενότητα 6.2).

Στην πρώτη περίπτωση, ο χρήστης επιλέγει το κουμπί AVAILABLE της κατηγορίας, στην κάρτα ενημέρωσης γνωρισμάτων, οπότε προτείνουμε ένα σύνολο από οντότητες, περιπτώσεις του πεδίου τιμών που έχει καθοριστεί. Από αυτές ο χρήστης μπορεί να διαλέξει κάποια, ως τιμή γνωρίσματος (σχήμα 4.17).



Σχήμα 4.17: Δημιουργία γνωρίσματος

Για την κατηγορία (**PhysicalObject**, *size*) επιλέγουμε το κουμπί AVAILABLE, και δημιουργούμε γνώρισμα με τιμή γνωρίσματος το **Medium**, επιλεγμένο από ένα σύνολο υπαρχουσών τιμών, περιπτώσεις της κλάσης **Size**. Φαίνεται η βοηθητική κάρτα (Input Card - Select Size) με τις προσφερόμενες επιλογές μεγεθών. Για την κατηγορία (**ArtificialObject**, *modified*) επιλέγουμε το κουμπί NEW, και δημιουργούμε γνώρισμα με τιμή γνωρίσματος το **ΑντικείμενοΓΕ11238'modified** το οποίο δημιουργείται σαν καινούργιος κόμβος στη βάση. Φαίνεται η βοηθητική κάρτα (Input Card - Instantiate class Modification), που μας πληροφορεί ότι ο κόμβος που δημιουργείται γίνεται περίπτωση της κλάσης **Modification**, η οποία αποτελεί το πεδίο τιμών γνωρίσματος. Η κατάσταση ορατότητας ονόματος γνωρισμάτων της κάρτας έχει τιμή HIDDEN LABEL, οπότε δεν ασχολούμαστε με τα ονόματα των γνωρισμάτων που θα δημιουργηθούν.

Στη δεύτερη περίπτωση, ο χρήστης επιλέγει το κουμπί NEW της κατηγορίας στην κάρτα ενημέρωσης γνωρισμάτων και ζητούμε να καθορίσει το όνομα του κόμβου που θέλει να δημιουργήσει. Η ταξινόμησή του γίνεται στο πεδίο τιμών που έχει καθοριστεί από πριν (σχήμα 4.17).

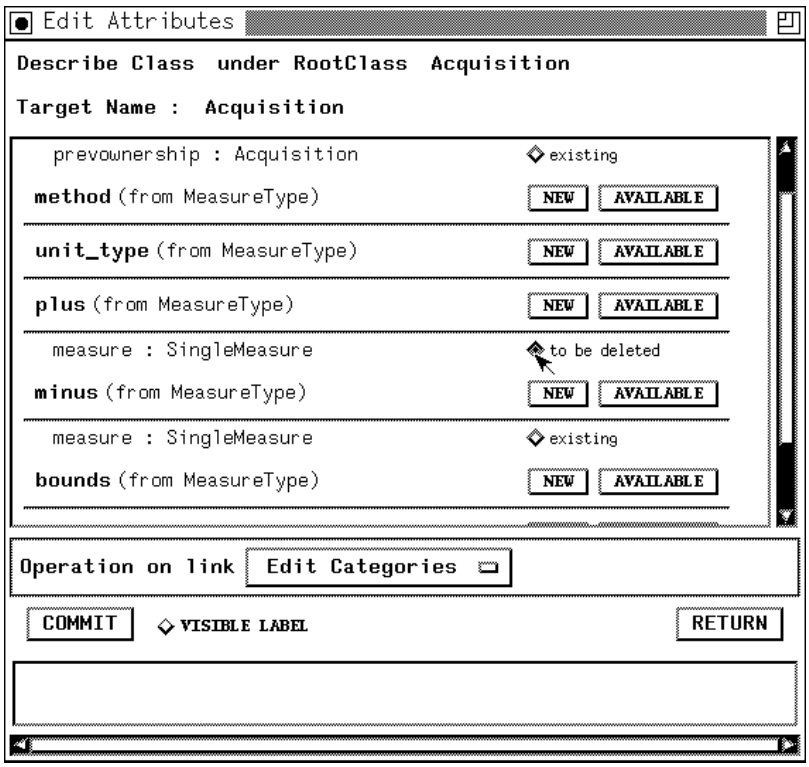
Αφού καθοριστεί η τιμή γνωρίσματος, ο χρήστης μπορεί να δώσει πληκτρολογώντας το όνομα του ίδιου του γνωρίσματος. Αυτό επιτρέπεται στην περίπτωση που η κατάσταση ορατότητας ονόματος γνωρισμάτων έχει τιμή VISIBLE LABEL. Διαφορετικά, το όνομα του γνωρίσματος αγνοείται.

Τελικά το γνώρισμα, μετά τον καθορισμό του, παρουσιάζεται στην κατάσταση <to be added>, ενδεικτικό ότι θα δημιουργηθεί με την πρώτη αίτηση για ενημέρωση (COMMIT).

4.5.6.4 Διαγραφή γνωρίσματος

Ομοια με τις κάρτες ταξινόμησης και γενίκευσης, στην κάρτα ενημέρωσης γνωρισμάτων η διαγραφή ενός γνωρίσματος γίνεται με χρήση των μεταβλητών κατάστασης <existing> - <to be deleted> που είδαμε και στην ενότητα 4.5.4. Κάνοντας την κατάσταση του γνωρίσματος <to be deleted>, αυτό διαγράφεται, με την πρώτη αίτηση για ενημέρωση (COMMIT).

Σημειώνεται ότι η διαγραφή ενός γνωρίσματος οδηγεί σε σβήσιμο μιας οντότητας από τη βάση. Συνεπώς το γνώρισμα διαγράφεται από όλες τις κατηγορίες που αυτό είναι περίπτωση, όπως φαίνεται στο σχήμα 4.18.



Σχήμα 4.18: Διαγραφή γνωρίσματος

Ακόμα κι αν το γνώρισμα είναι περίπτωση πολλών κλάσεων, αίτηση για διαγραφή του από μία κατηγορία, οδηγεί σε διαγραφή του από τη βάση, άρα κι από όλες τις κατηγορίες του. Ετσι αν ζητήσουμε να ενημερωθεί η βάση με τις αλλαγές που βλέπουμε στην εικόνα, τότε το γνώρισμα (**Acquisition**, *measure*), με τιμή το **SingleMeasure** διαγράφεται από τη βάση. Επομένως παύει να υπάρχει ως περίπτωση της κατηγορίας (**MeasureType**, *plus*) αλλά και της (**MeasureType**, *minus*).

Κεφάλαιο 5

Διατήρηση Περιορισμών

Το σύστημα σημασιολογικού ελέγχου του ΣΣΕ, όπως έχουμε ήδη αναφέρει, φροντίζει για τη διατήρηση δομικών περιορισμών στη βάση. Ομως έχουν παρουσιαστεί περιπτώσεις που οι έλεγχοι αυτοί δεν είναι αρκετοί και χρειάζονται επέκταση. Θέλοντας να υποστηρίξουμε συγκεκριμένους περιορισμούς ακεραιότητας που εξυπηρετούν ανάγκες εφαρμογών στις οποίες το ΔΕΔ χρησιμοποιήθηκε, έχουμε δημιουργήσει ένα μοντέλο περιορισμών. Δεν είναι σκοπός της παρούσας εργασίας η παρουσίαση ενός πλήρους μοντέλου διατήρησης περιορισμών. Μια εκτεταμένη μελέτη στο θέμα έχει γίνει στο [Θεο95a]. Επίσης περιορισμοί και έλεγχοι στις ενημερώσεις οντοτήτων προτείνονται και στο [Τζί95b], με τη χρήση ενός μετα-μοντέλου Οψεων Ενημέρωσης.

Το μοντέλο περιορισμών αποτελείται από ένα σύνολο οντοτήτων, τις οποίες το ΔΕΔ ερμηνεύει κατάλληλα και φροντίζει για τη διατήρηση των περιορισμών που συμβολίζουν. Οι οντότητες στις οποίες αναφερόμαστε είναι ο κόμβος **RestrictedVocabulary** σε επίπεδο μετα-κλάσης και οι μετα-κατηγορίες (**Individual**, *non_shared*), (**Individual**, *dependent*), (**Individual**, *necessary*) και (**Individual**, *interconnected*). Κάθε μια από αυτές τις οντότητες επιτελεί κάποιο ρόλο στην τήρηση κανόνων σημασιολογικού ελέγχου από το ΔΕΔ. Σε αυτές να προσθέσουμε την κατηγορία (**Update**, *sequence*) του ΜΔΕ, που βοηθά κι αυτή στην διατήρηση λογικών ακολουθιών λειτουργιών ενημέρωσης. Οι τελευταίες αποσκοπούν στη διατήρηση περιορισμών ακεραιότητας.

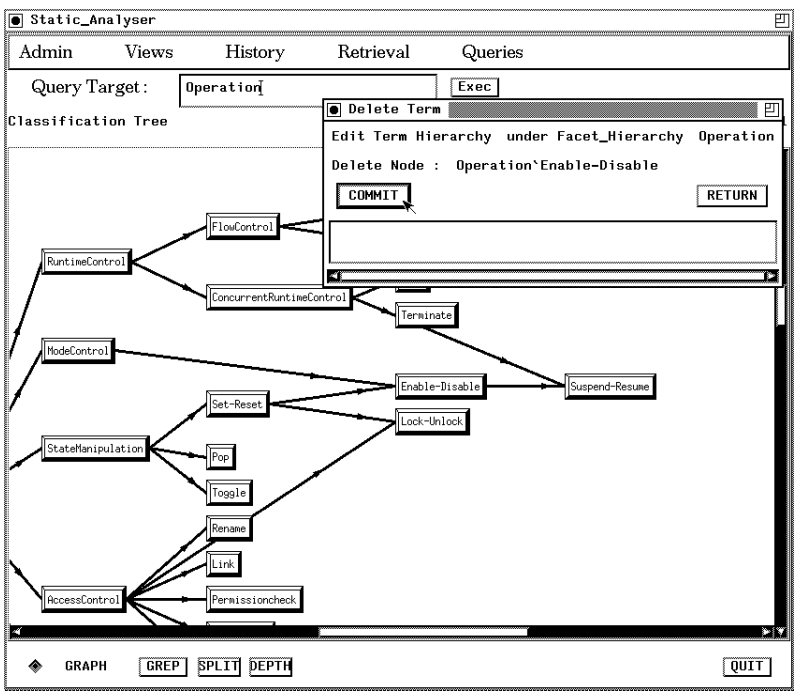
Στη συνέχεια του κεφαλαίου, περιγράφουμε τον τρόπο που το ΔΕΔ διατηρεί σημασιολογικούς περιορισμούς, με χρήση των παραπάνω οντοτήτων. Παραθέτουμε τις ανάγκες που μας ώθησαν στην επιβολή τέτοιων περιορισμών. Παρουσιάζουμε μηχανισμούς, όπως αυτός της διατήρησης σύνδεσης ιεραρχιών εξειδίκευσης, που επίσης φροντίζουν για την τήρηση σημασιολογικών περιορισμών στο μοντέλο εφαρμογής.

5.1 Διατήρηση σύνδεσης ιεραρχιών

Πολλές φορές παρουσιάζεται η ανάγκη να συνδέσουμε κόμβους με κάποιο είδος συνδέσμου, ώστε να φτιάξουμε ιεραρχίες. Η διατήρηση της σύνδεσης τέτοιων ιεραρχιών είναι σημαντική, αφού δε θέλουμε η αποκοπή ενός κόμβου μέλους της, να αλλοιώσει την ιεραρχία. Στο ΔΕΔ έχουν υλοποιηθεί μηχανισμοί που βοηθούν στη διατήρηση της σύνδεσης ιεραρχιών από κόμβους, όταν κάποιος από αυτούς σβηστεί. Στη συνέχεια περιγράφουμε τους μηχανισμούς για τα δύο είδη ιεραρχιών που χρησιμοποιούμε.

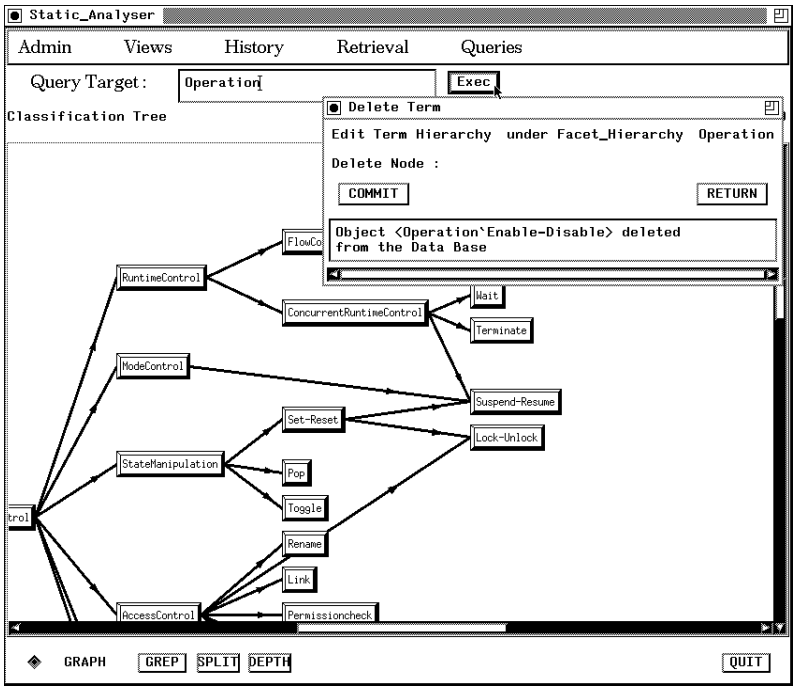
5.1.1 Ιεραρχίες εξειδίκευσης

Οι σύνδεσμοι εξειδίκευσης είναι οι συνηθέστεροι για την κατασκευή ιεραρχιών. Εχουμε υλοποιήσει ένα μηχανισμό ο οποίος κρατά συνδεδεμένους τους κόμβους που βρίσκονται σε μια τέτοια ιεραρχία. Ετσι, αν κάποιος κόμβος σβηστεί, τότε οι άμεσες υποκλάσεις του γίνονται υποκλάσεις των άμεσων υπερκλάσεών του, διατηρώντας τη σημασιολογία της ιεραρχίας. Ενα παράδειγμα βλέπουμε στα σχήματα 5.1 και 5.2



Σχήμα 5.1: Διαγραφή κόμβου από ιεραρχία εξειδίκευσης. Κατάσταση πριν τη διαγραφή.

Ο κόμβος **Operation‘Enable-Disable** βρίσκεται σε μια ιεραρχία υλοποιημένη με συνδέσμους εξειδίκευσης. Βλέπουμε τη θέση του στο δένδρο της ιεραρχίας πριν τη διαγραφή του, με τη βοήθεια του γραφικού αναλυτή, ενώ από το ΔΕΔ ζητούμε τη διαγραφή του.



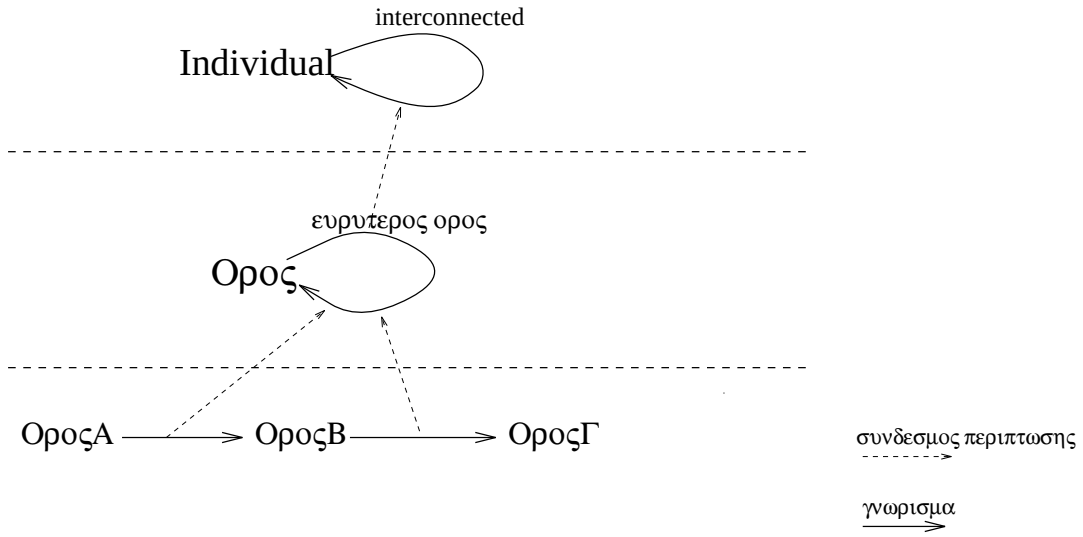
Σχήμα 5.2: Διαγραφή κόμβου από ιεραρχία εξειδίκευσης. Κατάσταση μετά τη διαγραφή.

Μετά τη διαγραφή του **Operation‘Enable-Disable**, μηχανισμός του ΔΕΔ ενώνει τις άμεσες υποκλάσεις του (**Operation‘Suspend-Resume**) με τις άμεσες υπερκλάσεις του (**Operation‘ModeControl** και **Operation‘Set-Reset**).

5.1.2 Ιεραρχίες γνωρισμάτων

Πολλές φορές υπάρχει η ανάγκη να διατηρείται η σύνδεση σε ορισμένες ιεραρχίες γνωρισμάτων, όπως ακριβώς συμβαίνει και στις ιεραρχίες εξειδίκευσης. Εχουμε εισάγει στο μοντέλο περιορισμών την μετα-κατηγορία *interconnected* που εκκινεί από το **Individual** και έχει τιμή γνωρίσματος το **Individual**. Κάθε κατηγορία γνωρίσματος που είναι περίπτωση της *interconnected*, αποκτά την ιδιότητα να διατηρεί συνδεδεμένους τους κόμβους που συνδέονται με γνωρίσματα περιπτώσεις της.

Ενα τέτοιο παράδειγμα έχουμε στο μοντέλο AAT που περιγράφει θησαυρό όρων. Κάθε όρος έχει ένα σύνδεσμο *ευρύτερος όρος* προς κάποιον άλλον. Αν θεωρήσουμε την κατάσταση που φαίνεται στο σχήμα 5.3 , όπου ο **ΟροςΓ** είναι ευρύτερος από τον **ΟροςΒ** που με τη σειρά του είναι ευρύτερος από τον **ΟροςΑ**, τότε διαγραφή του **ΟροςΒ** θα χαλούσε τη σύνδεση των όρων που βρίσκονται κάτω από την ‘‘ιεραρχία’’ των ευρύτερων όρων. Αν κάνουμε την κατηγορία *ευρύτερος όρος* περίπτωση της *interconnected*, τότε αυτή αποκτά την ιδιότητα να διατηρεί συνδεδεμένους τους κόμβους που συνδέονται με γνωρίσματά της. Οταν λοιπόν πρόκειται να σβηστεί ένας κόμβος που ανήκει σε μια τέτοια ιεραρχία, συνδέουμε τους κόμβους που τον αναφέρουν, με τους κόμβους που αυτός αναφέρει (εννοείται ότι η αναφορά και η σύνδεση είναι με γνωρίσματα της εν λόγω ιεραρχίας).



Σχήμα 5.3: Συνδεδεμένοι όροι μέσω της ''ιεραρχίας'' των ευρύτερων όρων

Διαγραφή του **ΟροςΒ** θα χαλούσε τη σύνδεση των όρων, κάτι που δεν είναι επιθυμητό. Επειδή όμως αυτός συνδέεται με γνωρίσματα κατηγορίας (**Ορος**, *ευρύτερος όρος*), που με τη σειρά της είναι περίπτωση της (**Individual**, *interconnected*), φροντίζουμε μετά τη διαγραφή του, να διατηρήσουμε τη σύνδεση του **ΟροςΑ** με τον **ΟροςΓ**.

5.2 Σύνολα αναλλοίωτου πληθυσμού

Οπως είδαμε στο προηγούμενο κεφάλαιο, στις κάρτες ενημέρωσης γνωρισμάτων, έχουμε δώσει τη δυνατότητα να δημιουργούμε κόμβους ως τιμές γνωρισμάτων, όταν κάτι τέτοιο είναι απαραίτητο. Αυτοί οι κόμβοι γίνονται συνήθως περιπτώσεις κάποιας κλάσης αυξάνοντας τον πληθυσμό της. Επίσης κατά τη διαγραφή γνωρισμάτων, υπάρχει περίπτωση να διαγραφεί και η τιμή γνωρίσματος (δες ενότητα 5.3).

Ομως υπάρχουν κλάσεις οι οποίες απαιτούμε να έχουν συγκεκριμένα και μόνο αντικείμενα ως περιπτώσεις τους. Χρησιμοποιούμε τη μετα-κλάση **RestrictedVocabulary**, για να ταξινομήσουμε σε αυτήν κάθε κλάση που θέλουμε να έχει την παραπάνω ιδιότητα. Ετσι αν θελήσουμε να αποδώσουμε σε μια οντότητα τιμή γνωρίσματος που να ανήκει σε κόμβο περίπτωση της **RestrictedVocabulary**, το ΔΕΔ το αντιλαμβάνεται και προτείνει μόνο τις υπάρχουσες οντότητες. Δηλαδή δεν επιτρέπεται, από την κάρτα ενημέρωσης γνωρισμάτων, η δημιουργία καινούργιας οντότητας ως περίπτωσης κλάσης που ανήκει στην **RestrictedVocabulary**. Επίσης, κατά τη διαγραφή γνωρίσματος, αποκλείεται η διαγραφή της τιμής γνωρίσματος, αν αυτή ανήκει σε κλάση, περίπτωση της μετα-κλάσης **RestrictedVocabulary**. Βλέπουμε λοιπόν ότι σύνολα αναλλοίωτου πληθυσμού υποστηρίζονται από την κάρτα ενημέρωσης γνωρισμάτων[1].

Για παράδειγμα στο ΚΛΕΙΩ έχει οριστεί η κλάση Month, που έχει ως περιπτώσεις της τους δώδεκα μήνες του χρόνου. Δε θα θέλαμε από λάθος να αλλάξει η σύνθεση αυτού του συνόλου με την

[1]Αλλοίωση του πληθυσμού κλάσεων μπορεί να γίνει επίσης από τις κάρτες δημιουργίας κόμβου και διαγραφής κόμβου. Ομως σε αυτές, με χρήση του συνόλου ορισμάτων τους, μπορούμε, από το μοντέλο διεργασιών, να αποτρέψουμε ανεπιθύμητες αλλοιώσεις πληθυσμού κλάσεων.

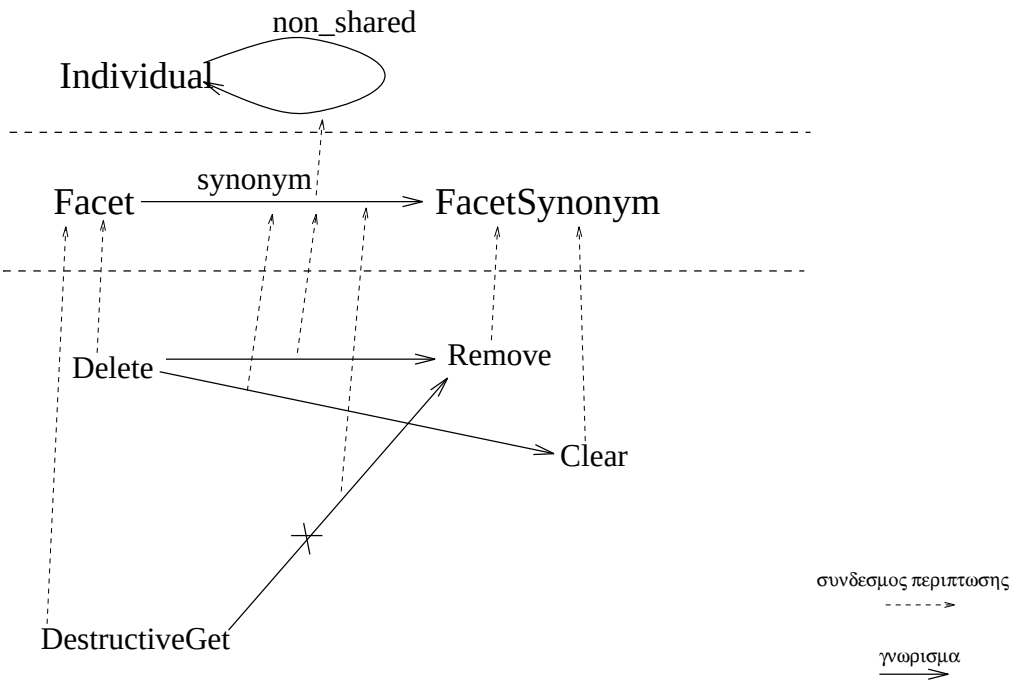
προσθήκη καινούργιου αντικειμένου ή τη διαγραφή κάποιου. Ετσι ταξινομούμε αυτήν την κλάση στην **RestrictedVocabulary**. Παρόμοιες κλάσεις είναι οι Size και Direction από την ίδια εφαρμογή.

5.3 Μη κοινά γνωρίσματα

Το μοντέλο δεδομένων της Telos επιτρέπει σε κόμβους να αναφέρονται σαν τιμή γνωρίσματος από περισσότερες από μία οντότητες. Αυτό όμως δεν είναι πάντα επιθυμητό. Θέλουμε πολλές φορές ένα γνώρισμα να είναι μοναδικό για μια οντότητα.

Για το σκοπό αυτό χρησιμοποιούμε τη μετα-κατηγορία *non_shared*, που εκκινεί από το **Individual** και έχει τιμή γνωρίσματος το **Individual**. Σε αυτήν ταξινομούμε κατηγορίες γνωρισμάτων για τις οποίες δε θέλουμε οι περιπτώσεις τους να έχουν κοινή τιμή γνωρίσματος, όταν εκκινούν από διαφορετικούς κόμβους. Ετσι, όταν πρόκειται να δημιουργήσουμε γνωρίσματα τέτοιων κατηγοριών, οι τιμές γνωρίσματος που επιτρέπονται από το ΔΕΔ είναι κόμβοι που δημιουργούνται επί τούτου ή κόμβοι που δεν αποτελούν τιμή γνωρίσματος για κανένα γνώρισμα της ίδιας κατηγορίας.

Ενα τέτοιο παράδειγμα έχουμε στο μοντέλο SIB Class Management System, όπου ένας όρος (Facet) μπορεί να έχει ένα ή περισσότερα συνώνυμα (FacetSynonym), αλλά κανένα συνώνυμο δεν μπορεί να ανήκει ταυτόχρονα σε δύο όρους (δες σχήμα 5.4).



Σχήμα 5.4: (**Facet**, *synonym*) : περίπτωση της μετα-κατηγορίας (**Individual**, *non_shared*)

Ενα συνώνυμο (**FacetSynonym**) μπορεί να ανήκει σε έναν μόνο όρο **Facet**. Απαγορεύουμε σε οποιοδήποτε όρο να δεχθεί σαν συνώνυμο κάποιο που ήδη αναφέρεται από άλλον όρο. Ετσι το **DesctructiveGet** δεν μπορεί να έχει συνώνυμο το **Remove**, αφού αυτό ήδη αναφέρεται σαν συνώνυμο από τον όρο **Delete**.

Σε αυτήν ακριβώς την περίπτωση των μη κοινών γνωρισμάτων, όταν, από την κάρτα ενημέρωσης

γνωρισμάτων, πρόκειται να σβήσουμε ένα σύνδεσμο που ανήκει σε κατηγορία περίπτωση της *non_shared*, είναι δυνατό να σβηστεί **και** ο κόμβος που αποτελεί τιμή γνωρίσματός του. Αυτό βέβαια θα συμβεί, αν ο κόμβος που αποτελεί την τιμή γνωρίσματος επιτρέπεται να διαγραφεί από τη βάση (δηλαδή δεν έχει περιπτώσεις, δεν έχει γνωρίσματα που εκκινούν από αυτόν, ούτε γνωρίσματα που καταλήγουν σε αυτόν[2]).

5.4 Γνωρίσματα απαραίτητα για την ύπαρξη κόμβων

Στο μοντέλο δεδομένων της Telos, δεν επιβάλλεται στις οντότητες, που έχουν κατηγορίες γνωρισμάτων, να έχουν υποχρεωτικά γνωρίσματα για αυτές. Δεν υπάρχει κανένας περιορισμός πλήθους στο θέμα αυτό.

Για κάποιες οντότητες όμως υπάρχουν γνωρίσματα τα οποία χαρακτηρίζονται απαραίτητα. Για να εκφράσουμε μια τέτοια απαίτηση, χρησιμοποιούμε τη μετα-κατηγορία *necessary*, που εκκινεί από το **Individual** και έχει τιμή γνωρίσματος το **Individual**. Σε αυτήν ταξινομούμε κατηγορίες για τις οποίες πρέπει υποχρεωτικά να δώσουμε γνωρίσματα. Ετσι, κάθε οντότητα που έχει κατηγορία γνωρισμάτων περίπτωση της *necessary*, πρέπει να έχει τουλάχιστο ένα γνώρισμα για αυτήν.

Για παράδειγμα στην εφαρμογή ΑΑΤ έχουμε θεωρήσει ότι ένας όρος έχει απαραίτητα έναν ευρύτερο[3], δηλώνοντας ότι η κατηγορία *ευρύτερος όρος* είναι περίπτωση της μετα-κατηγορίας *necessary*. Για τη διεργασία που αναλαμβάνει την ενημέρωση όρων, εφαρμόζονται τα παρακάτω:

- Κατά τη δημιουργία ενός όρου ζητείται τουλάχιστο ένας ευρύτερός του κάτω από τον οποίο θα ‘‘ταξινομηθεί’’. Αυτό επιτυγχάνεται με χρήση του γνωρίσματος (**Update**, *sequence*) του ΜΔΕ, όπου απαιτούμε από τη λειτουργία που υλοποιεί τη δημιουργία όρου, να ακολουθείται από μια λειτουργία που ενημερώνει τα γνωρίσματά του.

- Προσπάθεια να ενημερώσουμε τα γνωρίσματα ενός όρου, είναι επιτυχής μόνο αν έχουμε αποδόσει σε αυτόν ένα τουλάχιστον ευρύτερο όρο.

- Από την κάρτα ενημέρωσης γνωρισμάτων δεν επιτρέπεται διαγραφή όλων των γνωρισμάτων που συνδέουν έναν όρο με τους ευρύτερούς του. Ομως αν μια οντότητα έχει γνωρίσματα, δεν μπορεί να διαγραφεί, σύμφωνα με το σύστημα σημασιολογικού ελέγχου της Telos. Ετσι, αν από την κάρτα διαγραφής κόμβου αιτήσουμε διαγραφή ενός όρου, από τον οποίο εκκινούν τέτοια γνωρίσματα, έχουμε σαν αποτέλεσμα τη διαγραφή όχι μόνο του όρου αλλά και των γνωρισμάτων αυτών.

[2]Το γνώρισμα που καταλήγει σε αυτόν, με οντότητα εκκίνησης την ΕΟ της κάρτας δε λαμβάνει μέρος σε αυτόν τον υπολογισμό, αφού έχουμε ήδη ζητήσει τη διαγραφή του.

[3]Η εφαρμογή έχει έναν ενσωματωμένο (built-in) όρο NULLTerm, τον οποίο έχουν σαν ευρύτερο όλες οι κορυφές των ιεραρχιών.

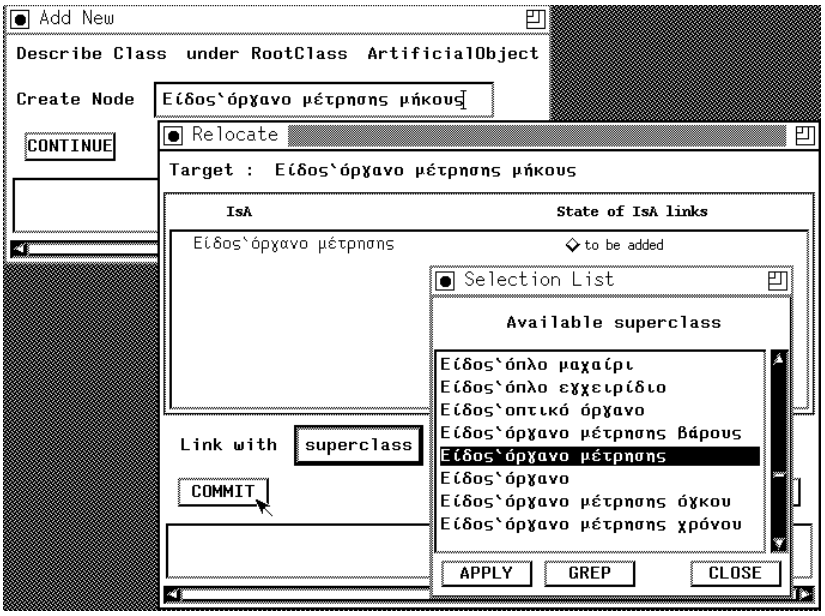
Θεωρώντας αυτή τη μετα-κατηγορία σαν σχέση που συνδέει τον κόμβο από τον οποίο ξεκινά και αυτόν στον οποίο καταλήγει, μπορούμε να πούμε ότι είναι μια υλοποίηση της ολικής σχέσης, όπως αυτή εμφανίζεται στο [Θεο95a]. Εκεί αναφέρεται ότι μια σχέση από το Α στο Β είναι ολική αν κάθε οντότητα τύπου Α συνδέεται με μια τουλάχιστο οντότητα τύπου Β.

5.5 Ακολουθίες λειτουργιών

Το μοντέλο διεργασιών, όπως περιγράφηκε στην ενότητα 3.2, μας παρέχει τη δυνατότητα να έχουμε ακολουθίες από λειτουργίες ενημέρωσης. Ετσι μια λειτουργία μπορεί να ακολουθείται από μια δεύτερη, η οποία θα δράσει στην ΕΟ της πρώτης. Στις εφαρμογές στις οποίες το ΔΕΔ έχει χρησιμοποιηθεί, η πιο συνηθισμένη περίπτωση που απαιτούμε μια λειτουργία να ακολουθείται από μια δεύτερη, είναι όταν η πρώτη είναι λειτουργία δημιουργίας κόμβου ενώ η δεύτερη είναι, ανάλογα με την περίπτωση, λειτουργία ταξινόμησης, εξειδίκευσης ή ενημέρωσης γνωρισμάτων. Ετσι ζητούμε από το σύστημα να ταξινομήσει, να εξειδικεύσει ή να ενημερώσει τα γνωρίσματα του κόμβου που πρόκειται να δημιουργηθεί σε μια δοσοληψία, με σκοπό να διατηρηθεί η σημασιολογική ορθότητα του μοντέλου εφαρμογής.

Η κάρτα που αντιστοιχεί σε λειτουργία που, στο μοντέλο διεργασιών, ακολουθείται από άλλη, δεν εμφανίζει την επιλογή COMMIT, αλλά την αντικαθιστά με την επιλογή CONTINUE, ενδεικτικό αυτής της κατάστασης. Με την επιλογή CONTINUE, η ακόλουθη λειτουργία υλοποιείται με κατάλληλη κάρτα ενημέρωσης, ενώ η ΕΟ της πρώτης γίνεται ΕΟ και της δεύτερης κάρτας, όπως φαίνεται στο σχήμα 5.5. Στις περιπτώσεις που εμφανίζονται ακολουθίες καρτών αυτού του είδους, θέλουμε να ενημερώσουμε την ΕΟ (που είναι κοινή για όλες τις κάρτες της ακολουθίας), με πράξεις που δεν προσφέρονται από μια μόνο λειτουργία. Επσης δε θα θέλαμε να γίνουν δεκτές οι ενημερώσεις κάποιας λειτουργίας, όταν κάποια άλλη της ακολουθίας αποτύχει, αφού κάτι τέτοιο θα οδηγούσε σε πιθανά σημασιολογικά λάθη. Ετσι τη δικαιοδοσία για ενημέρωση της βάσης με τις αλλαγές που όλες οι κάρτες τις ακολουθίας επιβάλλουν, έχει η τελευταία κάρτα της ακολουθίας.

Για παράδειγμα, στο ΚΛΕΙΩ, κατά τη δημιουργία κλάσης, απαιτούμε ταυτόχρονα την ένταξή της σε κάποια από τις ιεραρχίες εξειδίκευσης κλάσεων του μοντέλου εφαρμογής. Για να το πετύχουμε αυτό, ορίζουμε στα πλαίσια μιας διεργασίας στο μοντέλο διεργασιών του ΔΕΔ για το ΚΛΕΙΩ, η λειτουργία *Add_New*, που είναι υπεύθυνη για τη δημιουργία κλάσεων, να ακολουθείται από μια λειτουργία *Relocate*, που τοποθετεί μια κλάση σε ιεραρχία εξειδίκευσης. Παράδειγμα ακολουθίας λειτουργιών καθορισμένη από το μοντέλο διεργασιών για το ΚΛΕΙΩ, παρουσιάζεται στο σχήμα 5.5. Αν δεν είχαν υλοποιηθεί αυτές οι δύο λειτουργίες, σαν μία σύνθετη ορισμένη στο μοντέλο διεργασιών, τότε θα μπορούσε να είχε επιτύχει η πρώτη, η δημιουργία του κόμβου, ενώ να είχε αποτύχει η δεύτερη. Τότε ένας βασικός κανόνας του μοντέλου εφαρμογής, η ένταξη κάθε κλάσης σε μια ιεραρχία εξειδίκευσης κλάσεων, θα παραβιαζόταν.



Σχήμα 5.5: Ακολουθία λειτουργιών καθορισμένη από το μοντέλο διεργασιών

Με την επιλογή CONTINUE της κάρτας που υλοποιεί τη λειτουργία *Add_New*, εμφανίζεται η *Relocate* που είναι η ακόλουθή της λειτουργία, όπως έχει καθοριστεί στο μοντέλο διεργασιών.

Ενα άλλο παράδειγμα έχουμε στην εφαρμογή AAT, όπου έχει εισαχθεί η σχέση ‘‘ευρύτερος όρος’’. Η σχέση αυτή έχει ταξινομηθεί, όπως έχουμε ήδη εξηγήσει, στην κατηγορία γνωρισμάτων (**Individual**, *necessary*). Ετσι κατά τη δημιουργία ενός όρου, απαιτούμε απόδοση γνωρίσματος κατηγορίας (**Ορος**, *ευρύτερος όρος*) σε αυτόν. Για να το πετύχουμε αυτό, ορίζουμε[4] η λειτουργία *CreateTerm* να ακολουθείται από μια λειτουργία *DefineBroaderTerm*. Η μεν πρώτη είναι τύπου CREATE_NODE και είναι υπεύθυνη για τη δημιουργία όρων ενώ η δεύτερη είναι τύπου UPDATE_ATTRIBUTES και προσφέρει την κατηγορία *ευρύτερος όρος* για ενημέρωση.

[4]στο μοντέλο διεργασιών του ΔΕΔ για το AAT και συγκεκριμένα για τη διεργασία που αναλαμβάνει ενημέρωση όρων

Κεφάλαιο 6

Αυτομ ατισμοί και διευκόλυνση του χρήστη

Βασικός στόχος του ΔΕΔ είναι η διευκόλυνση του χρήστη, στην προσπάθειά του για ενημέρωση της βάσης. Για αυτό το λόγο, ένα σύνολο από κατηγορίες έχουν εισαχθεί στο μοντέλο που καθορίζει τη λειτουργικότητα του ΔΕΔ. Αυτές έχουν σκοπό τη διευκόλυνση του χρήστη. Το τμήμα αυτό του μοντέλου θα ονομάζουμε μοντέλο καθοδήγησης. Αποτελείται από τις κατηγορίες (**Individual**, *dependent*) και (**Individual**, *default_ef*), οι οποίες έχουν ειδική σημασία, όπως θα φανεί στη συνέχεια του κεφαλαίου. Επιπλέον, έχει αναπτυχθεί ένα σύνολο αυτομ ατισμών για προσφορά βοήθειας προς το χρήστη. Τέτοιοι είναι η ονομ ατοδοσία εξαρτημένων οντοτήτων, η ενημέρωση σύνθετων οντοτήτων, ο καθορισμός του πεδίου τιμών γνωρίσματος και η δυνατότητα επέκτασής του, τους οποίους παρουσιάζουμε στη συνέχεια του κεφαλαίου.

6.1 Σύνθετες οντότητες

Από την εφαρμογή του ΔΕΔ στο ΚΛΕΙΩ, παρουσιάστηκε η ανάγκη της ενημέρωσης σύνθετων οντοτήτων. Αυτές αποτελούν μια επέκταση του βασικού μοντέλου των οντοκεντρικών βάσεων δεδομένων, η οποία απαντάται αρκετά συχνά. Στο [Kim90] δίνεται ο εξής ορισμός:

> ‘‘Μια *σύνθετη οντότητα* είναι, εν γένει, μια συλλογή από ετερογενείς οντότητες διαφορετικών κλάσεων. Οι κλάσεις αυτές ονομάζονται *συνιστώσες κλάσεις* και ορίζουν μια ιεραρχία που ονομάζεται *ιεραρχία σύνθετων γνωρισμάτων*.’’

Σε ορισμένα συστήματα (όπως το ORION [Kim90]), υπάρχει η έννοια της σύνθετης οντότητας.

Στην Telos δεν υπάρχει αυτή η έννοια. Δεν υπάρχουν ‘‘όρια’’ στον ορισμό μιας οντότητας, όσον αφορά στο βάθος ανάλυσης των γνωρισμάτων της. Μπορούμε όμως να θεωρήσουμε σύνθετες οντότητες αυτές που αποτελούνται από *εξαρτημένες οντότητες*, όπως αυτές ορίζονται στη συνέχεια.

Πριν περάσουμε στον ορισμό των εξαρτημένων οντοτήτων, ας δούμε τι σημαίνει σχέση εξάρτησης. Σαν σχέση εξάρτησης ορίζουμε τη διμελή σχέση μεταξύ κόμβων της βάσης (έστω Α και Β), που υποδεικνύει ότι ο Β εξαρτάται με κάποιο τρόπο από τον Α. Με αυτό συχνά εννούμε ότι ο Β αποτελεί μέρος του Α (π.χ. η σχέση των δεικτών του ρολογιού με το ρολόι), ή γενικότερα ότι ο Β συνδέεται στενά με τον Α (π.χ. ένα γεγονός συνδέεται με το χρόνο που συνέβει, ένα εύρημα συνδέεται με τον τόπο ανεύρεσής του). Η σχέση αυτή, που συμβολίζουμε (Α,Β), αποτελεί για την Telos ένα γνώρισμα. Για την παράστασή της έχει εισαχθεί, στο μοντέλο καθοδήγησης, η μετα-κατηγορία *dependent* που εκκινεί από **Individual** και έχει τιμή γνωρίσματος το **Individual**. Η σχέση εξάρτησης χαρακτηρίζεται από το γεγονός ότι ανήκει σε κατηγορία που είναι περίπτωση της (**Individual**, *dependent*). Κόμβος που βρίσκεται στο δεύτερο μέλος μια σχέσης εξάρτησης καλείται εξαρτημένη οντότητα. Για παράδειγμα, στη σχέση (Α,Β) που είδαμε προηγουμένως, η Β ονομάζεται εξαρτημένη οντότητα από την Α.

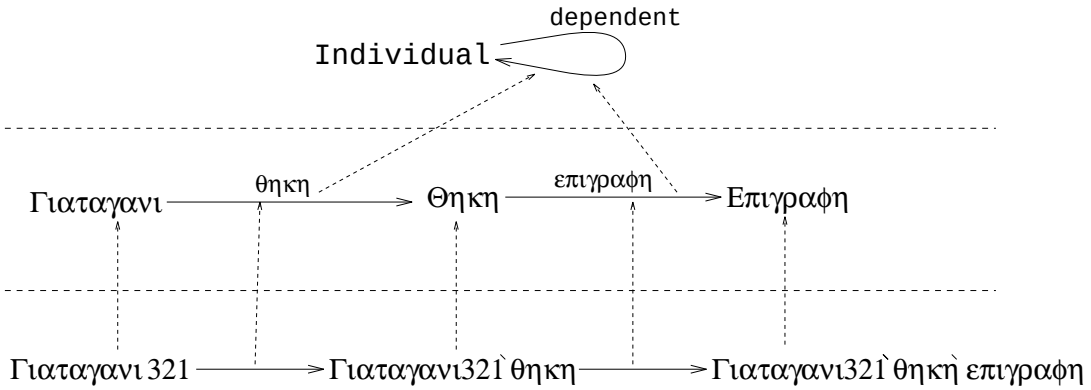
Σύνθετη ονομάζεται μια οντότητα όταν βρίσκεται στο πρώτο μέλος μιας σχέσης εξάρτησης ή, γενικότερα, όταν συνδέεται με σχέση εξάρτησης με μια σύνθετη οντότητα. Είναι προφανές ότι ορίζονται ιεραρχίες από σύνθετες οντότητες, όπου μια σύνθετη μπορεί να περιέχει άλλες σύνθετες κ.ο.κ. Σε ένα ανώτερο επίπεδο αφαίρεσης η σύνθετη οντότητα θα μπορούσε να θεωρηθεί μία και μόνο οντότητα.

6.1.1 Ονοματοδοσία εξαρτημένων οντοτήτων

Εχουμε ήδη αναφέρει ότι από την κάρτα ενημέρωσης γνωρισμάτων, μπορούμε να δημιουργούμε ένα καινούργιο κόμβο και αμέσως να τον συνδέουμε με την ΕΟ της κάρτας. Θα ήταν όμως επιθυμητό να προτείνεται κι ένα όνομα κάθε φορά για το καινούργιο αντικείμενο. Για το σκοπό αυτό υλοποιήθηκε μια τακτική ονοματοδοσίας με βάση μια παρατήρηση που προέκυψε από την εφαρμογή του ΔΕΔ στο ΚΛΕΙΩ: Οταν θέλουμε να συνδέσουμε έναν κόμβο με μια εξαρτημένη οντότητα (άρα η σύνδεσή τους γίνεται με σχέση εξάρτησης, δηλαδή με γνώρισμα που ανήκει σε κατηγορία, περίπτωση της *dependent*), τότε το όνομα της εξαρτημένης, μπορεί να κατασκευαστεί εύκολα, συναρτήσει του ονόματος του κόμβου εκκίνησης και του ονόματος της κατηγορίας του γνωρίσματος. Βέβαια όταν η κατηγορία γνωρίσματος είναι κάποιος πρωτογενής τύπος της Telos, τότε το προτεινόμενο όνομα δεν μπορεί να προβλεφθεί. Σε αυτή την περίπτωση δεν προτείνεται όνομα.

Εστω για παράδειγμα ότι αποδίδουμε στο **Γιαταγάνι321** (περίπτωση του **Γιαταγάνι**) γνώρισμα κατηγορίας *θήκη* που είναι εξαρτημένο (*dependent*), σύμφωνα με το μοντέλο του ΚΛΕΙΩ. Τότε, όπως φαίνεται και στο σχήμα 6.1, το όνομα που προτείνεται για τιμή γνωρίσματος είναι το **Γιαταγάνι321'θήκη**. Είναι φυσιολογικό, με την τακτική ονοματοδοσίας που έχει χρησιμοποιηθεί σε αυτό το μοντέλο, να δοθεί αυτό το όνομα.

Για τα ονόματα των εξαρτημένων οντοτήτων, έχει προταθεί (στο [Θεο95a]) μηχανισμός που επιτρέπει κόμβους χωρίς ονόματα. Μέχρι όμως να υλοποιηθεί αυτή η πρόταση, αρκούμαστε στο να διευκολύνουμε το χρήστη στην ονοματοδοσία των εξαρτημένων οντοτήτων με την παραπάνω



Σχήμα 6.1: Καθορισμός ονόματος εξαρτημένης οντότητας

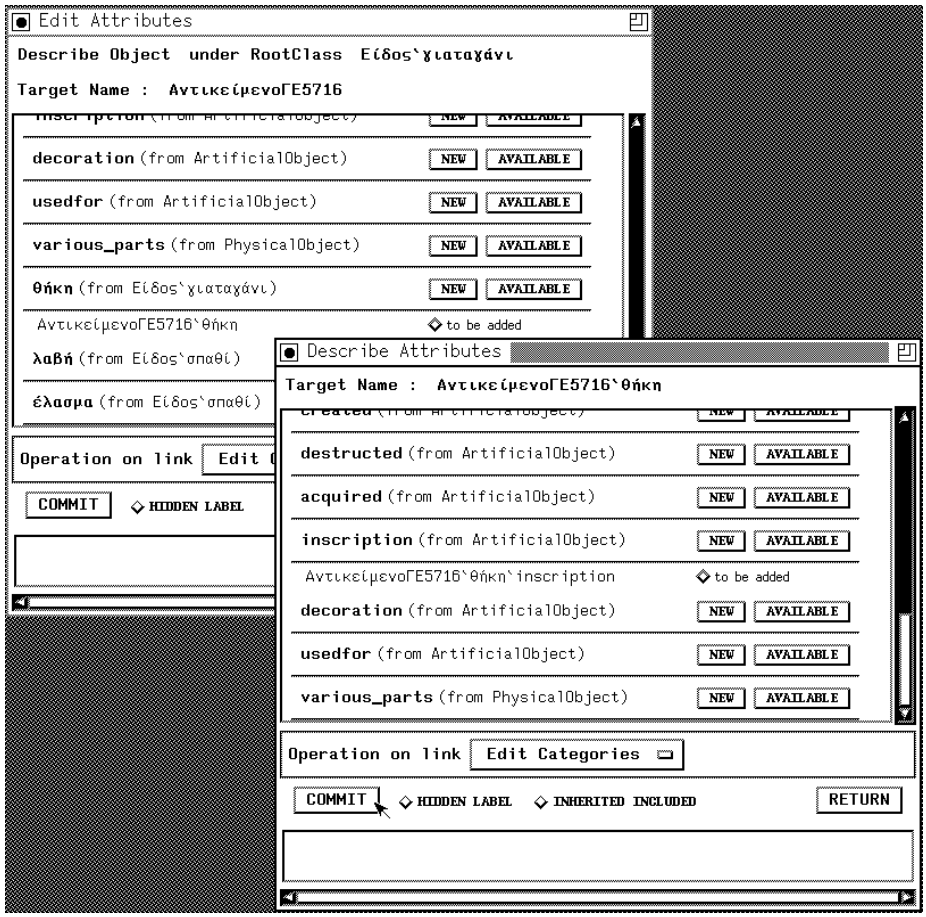
μέθοδο.

6.1.2 Ενημέρωση σύνθετων οντοτήτων

Για να γίνει εύκολη η εισαγωγή σύνθετων οντοτήτων, μπορούμε να κάνουμε χρήση της αναδρομικής κάρτας ενημέρωσης γνωρισμάτων. Η τελευταία είναι μια ειδική περίπτωση της κάρτας ενημέρωσης γνωρισμάτων, που καλείται από κάρτες ενημέρωσης γνωρισμάτων σύνθετων οντοτήτων.

Για κάθε γνώρισμα μιας σύνθετης οντότητας, που υλοποιεί σχέση εξάρτησης, μπορούμε να βρούμε την τιμή γνωρίσματός του και να ενημερώσουμε τα γνωρίσματα αυτής. Η ενημέρωση γίνεται με την εμφάνιση αναδρομικής κάρτας ενημέρωσης γνωρισμάτων. Αν με τη σειρά της η νέα κάρτα έχει ΕΟ σύνθετη οντότητα, τότε μπορούμε να ανοίξουμε νέα αναδρομική κάρτα ενημέρωσης γνωρισμάτων για τις εξαρτημένες από αυτήν οντότητες, κ.ο.κ. Ετσι, είναι φανερό ότι μπορούμε να ενημερώνουμε αλυσίδες εξαρτημένων οντοτήτων, σε όσο βάθος θέλουμε. Με αυτόν τον τρόπο υλοποιείται μια νέα μορφή αλυσίδας καρτών ενημέρωσης. Οπως και στις προηγούμενες μορφές αλυσίδων καρτών ενημέρωσης, έτσι κι εδώ, η τελευταία κάρτα της αλυσίδας έχει τη δικαιοδοσία να αιτήσει ενημέρωση (COMMIT) της βάσης με τις αλλαγές όλων των καρτών ενημέρωσης. Επειδή η ενημέρωση όλων των αναδρομικών καρτών ενημέρωσης γίνεται σε μία δοσοληψία, δίνεται η αίσθηση ότι πράγματι ενημερώνουμε σύνθετη οντότητα (δες σχήμα 6.2).

Για να κάνουμε εύκολη τη διαγραφή σύνθετων οντοτήτων, ακολουθούμε την αντίστροφη πορεία, από αυτήν της δημιουργίας τους. Συγκεκριμένα, για κάθε εξαρτημένη οντότητα που εμφανίζεται σε κατάσταση <existing>, στην κάρτα ενημέρωσης γνωρισμάτων της οντότητας από την οποία εξαρτάται, μπορούμε να ανοίξουμε την αναδρομική κάρτα ενημέρωσης γνωρισμάτων της. Από αυτήν μπορούμε να διαγράψουμε τυχόν γνωρίσματά της ενώ δίνεται η δυνατότητα να διαγράψουμε και την ίδια την εστιακή οντότητα, με μια επιπρόσθετη επιλογή DELETE της αναδρομικής κάρτας, που είναι ενεργή μόνο όταν η εστιακή της οντότητα υπάρχει στη βάση (δηλαδή δεν πρόκειται για οντότητα προς δημιουργία). Ενα παράδειγμα διαγραφής εξαρτημένης οντότητας από την αναδρομική



Σχήμα 6.2: Ενημέρωση σύνθετης οντότητας

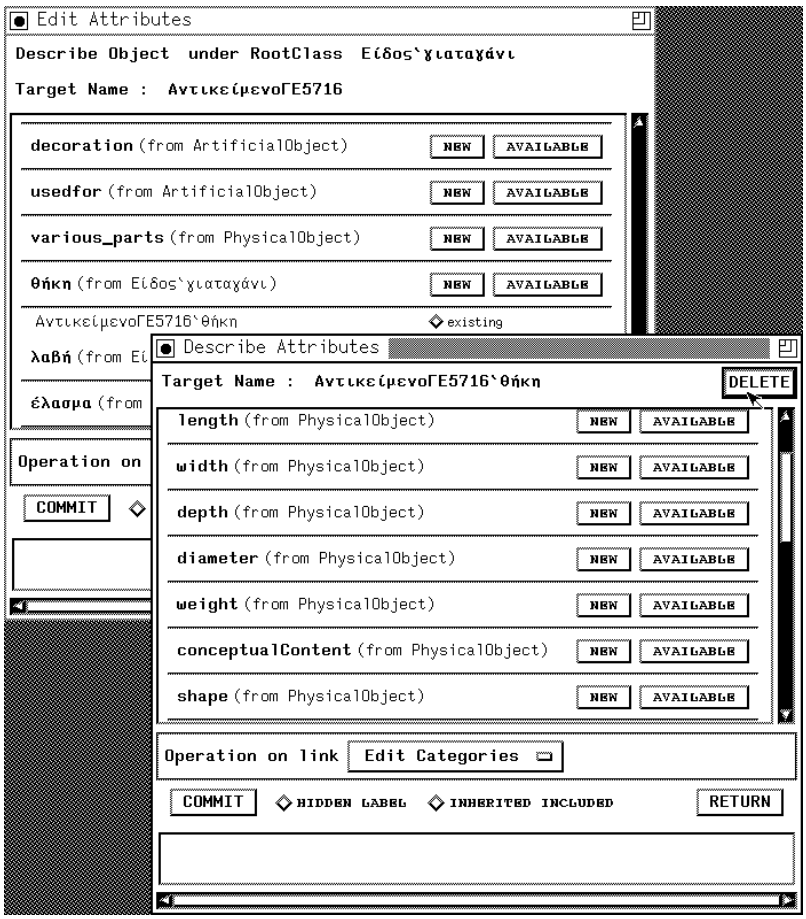
Το **ΑντικείμενοΓΕ5716** είναι περίπτωση της κλάσης **Είδος'γιαταγάνι**. Δημιουργούμε (κατάσταση <to be added>) τη θήκη του (ΑντικείμενοΓΕ5716'θήκη). Επειδή η τελευταία είναι εξαρτημένη οντότητα (η κατηγορία (**Είδος'γιαταγάνι**, *θήκη*) είναι περίπτωση της (**Individual**, *dependent*)), μπορούμε να εμφανίσουμε αναδρομική κάρτα ενημέρωσης γνωρισμάτων. Δημιουργούμε (κατάσταση <to be added>) την επιγραφή του (ΑντικείμενοΓΕ5716'θήκη'inscription). Με την αίτηση για ενημέρωση της βάσης από την τελευταία κάρτα, όλες οι ενημερώσεις γίνονται σε μια δοσοληψία δεκτές ή απορρίπτονται. Ετσι το **ΑντικείμενοΓΕ5716** ενημερώνεται σαν σύνθετη οντότητα.

κάρτα ενημέρωσης γνωρισμάτων της, βλέπουμε στο σχήμα 6.3. Με αυτήν την πρόσθετη επιλογή, διευκολύνουμε το έργο της διαγραφής των εξαρτημένων οντοτήτων και κατά συνέπεια κάνουμε ευκολότερη τη διαγραφή τμημάτων μιας σύνθετης οντότητας.

6.2 Καθορισμός πεδίου τιμών γνωρίσματος

Στις περιπτώσεις που θέλουμε να δημιουργήσουμε ένα γνώρισμα που εκκινεί από την ΕΟ της κάρτας ενημέρωσης γνωρισμάτων, πρέπει να δώσουμε την τιμή γνωρίσματός του, είτε επιλέγοντας από ένα σύνολο υπαρχόντων κόμβων είτε δημιουργώντας έναν καινούργιο για το σκοπό αυτό (δες ενότητα 4.5.6.3).

Οι κόμβοι αυτοί ανήκουν (σαν άμεσες ή έμμεσες περιπτώσεις) στην κλάση που αποτελεί τιμή



Σχήμα 6.3: Διαγραφή εξαρτημένης οντότητας

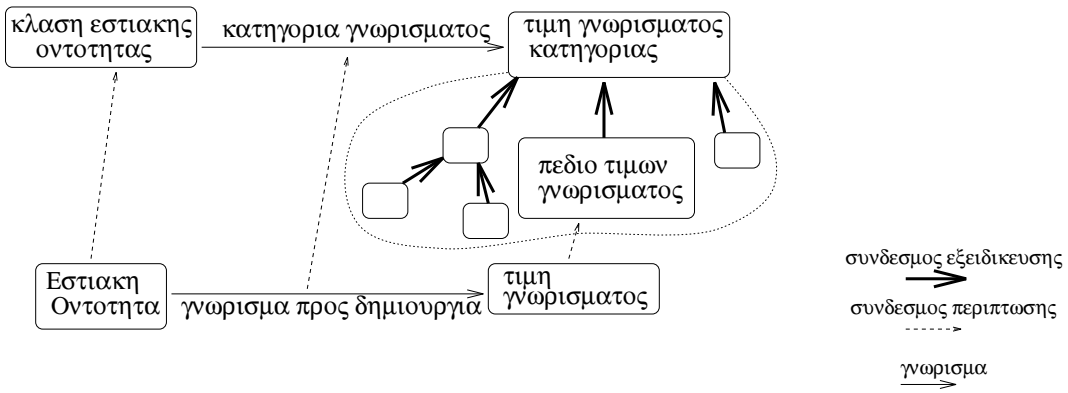
Το **ΑντικείμενοΓΕ5716'θήκη** είναι εξαρτημένη οντότητα από το **ΑντικείμενοΓΕ5716**. Μπορούμε από την αναδρομική κάρτα ενημέρωσης γνωρισμάτων της πρώτης, να ζητήσουμε διαγραφή της, με την επιλογή DELETE.

γνωρίσματος της κατηγορίας. Ομως πολλές φορές τυχαίνει αυτή να έχει υποκλάσεις, οπότε, κατά τον καθορισμό τιμής γνωρίσματος:

- αν επιλεγόταν υπάρχον κόμβος, ο χρήστης θα δυσκολευόταν να επιλέξει από τις ενδεχομένως πολυπληθείς άμεσες και έμμεσες περιπτώσεις της τιμής γνωρίσματος της κατηγορίας
- αν δημιουργούνταν καινούργιος κόμβος, ο χρήστης θα προβληματιζόταν με την άμεση ταξινόμησή του στην κορυφή της ιεραρχίας εξειδίκευσης, την τιμή γνωρίσματος της κατηγορίας

Για να διευκολύνουμε το χρήστη σε αυτή τη φάση, του επιτρέπουμε να καθορίζει το πεδίο τιμών γνωρίσματος, με μεγαλύτερη ακρίβεια. Σε κάθε περίπτωση βέβαια, για να είμαστε σύμφωνοι με το σύστημα σημασιολογικού ελέγχου του ΣΣΕ, πρέπει το πεδίο τιμών γνωρίσματος να είναι υποκλάση της τιμής γνωρίσματος της κατηγορίας[1], όπως φαίνεται και στο σχήμα 6.4.

[1]Για να είμαστε πιο ακριβείς, επιλογή πεδίου τιμών γνωρίσματος διαφορετικού από αυτό που η κατηγορία προτείνει, σημαίνει πως το γνώρισμα που δημιουργούμε θα μπορούσε να ανήκει σε **υποκατηγορία** της κατηγορίας που το ταξινομούμε.



Σχήμα 6.4: Οντότητα καθορισμού πεδίου τιμών γνωρίσματος

Εστω ότι για την <εστιακή οντότητα>, θέλουμε να δημιουργήσουμε το <γνώρισμα προς δημιουργία>, σαν περίπτωση της <κατηγορίας γνωρίσματος>. Η <τιμή γνωρίσματος κατηγορίας> μπορεί να έχει υποκλάσεις, μια από τις οποίες επιτρέπεται ο χρήστης να διαλέξει, σαν <πεδίο τιμών γνωρίσματος>. Ετσι η <τιμή γνωρίσματος> είτε γίνεται περίπτωση αυτής της κλάσης (στην περίπτωση δημιουργίας καινούργιου κόμβου για τιμή γνωρίσματος) είτε επιλέγεται από τις περιπτώσεις αυτής της κλάσης (στην περίπτωση επιλογής υπαρχόντων κόμβων για τιμή γνωρίσματος).

6.2.1 Καθορισμός από το χρήστη

Αν η τιμή γνωρίσματος της κατηγορίας έχει υποκλάσεις, τότε, πατώντας το αριστερό πλήκτρο του ποντικιού στα κουμπιά NEW ή AVAILABLE της κάρτας ενημέρωσης γνωρισμάτων, παρουσιάζεται στο χρήστη μια λίστα. Από αυτήν μπορεί να διαλέξει μία από όλες τις υποκλάσεις της τιμής γνωρίσματος της κατηγορίας, άμεσες και έμμεσες. Η επιλεγμένη κλάση αποτελεί το πεδίο τιμών γνωρίσματος. Αν βέβαια η τιμή γνωρίσματος της κατηγορίας δεν έχει υποκλάσεις, τότε αυτομάτως η ίδια επιλέγεται για πεδίο τιμών γνωρίσματος.

6.2.2 Καθορισμός από το μοντέλο

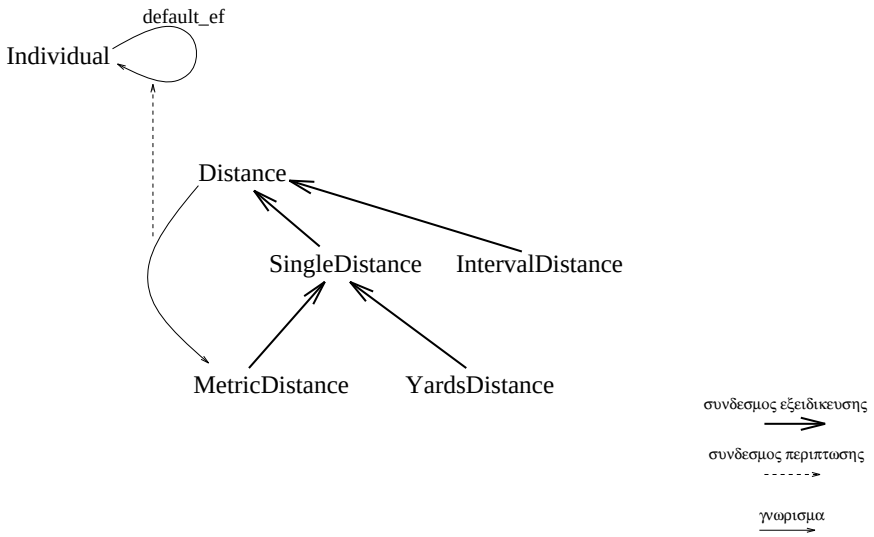
Από την εφαρμογή του ΔΕΔ στο ΚΛΕΙΩ, παρατηρήσαμε ότι είναι συνηθισμένο φαινόμενο για συγκεκριμένες κλάσεις που αποτελούν τιμή γνωρίσματος κατηγορίας, να επιλέγουμε, από τη λίστα των υποκλάσεων τους την ίδια οντότητα για κλάση καθορισμού πεδίου τιμών. Αυτό μας ώθησε να εισαγάγουμε στο μοντέλο την έννοια του προτεινόμενου πεδίου τιμών. Εχει εισαχθεί στο μοντέλο η μετα-κατηγορία *default_ef* που εκκινεί από **Individual** και έχει τιμή γνωρίσματος το **Individual**, ενώ χρησιμοποιείται ως εξής:

Εστω ότι μια κλάση Α είναι τιμή γνωρίσματος κατηγορίας γνωρισμάτων, κι έχει ένα σύνολο από υποκλάσεις. Αν υπάρχει υποκλάση της (έστω Β) που συνήθως επιλέγεται σαν πεδίο τιμών γνωρίσματος, τότε δημιουργούμε ένα γνώρισμα που να ανήκει στην κατηγορία *default_ef*, να εκκινεί από την Α και να έχει τιμή γνωρίσματος τη Β. Η Β αποτελεί το προτεινόμενο πεδίο τιμών της Α. Ο χρήστης μπορεί να ζητήσει το προτεινόμενο πεδίο τιμών με το πάτημα του δεξιού πλήκτρου του

Ομως στην πράξη η δημιουργία πολλών υποκατηγοριών δεν είναι πάντα επιθυμητή, αφού α) επιβαρύνουμε το μοντέλο και β) διαχωρίζουμε τις περιπτώσεις της τιμής γνωρίσματος της κατηγορίας, και δεν παρέχουμε σε αυτές ισότιμη μεταχείριση.

ποντικιού στα κουμπιά NEW ή AVAILABLE της κάρτας ενημέρωσης γνωρισμάτων. Αν και συνήθως το προτεινόμενο πεδίο τιμών είναι μοναδικό, μπορεί, σε μερικές περιπτώσεις, να αποδοθούν περισσότερα του ενός. Τότε, με τις ίδιες ενέργειες, παρουσιάζεται στο χρήστη λίστα με τις προτεινόμενες κλάσεις, για τελική επιλογή μιας. Αυτό που κερδίζουμε σε μια τέτοια περίπτωση είναι ότι, μικραίνοντας το πεδίο επιλογής, μπορούμε πιο γρήγορα να βρούμε το επιθυμητό.

Για παράδειγμα στο ΚΛΕΙΩ, η κλάση **MetricDistance** γίνεται προτεινόμενο πεδίο τιμών γνωρίσματος για την κλάση **Distance** (δες σχήμα 6.5). Ετσι όταν δημιουργούμε ένα γνώρισμα που η κατηγορία του έχει τιμή γνωρίσματος την **Distance**, η προτεινόμενη, από το ΔΕΔ, κλάση καθορισμού πεδίου τιμών γνωρίσματος, είναι η **MetricDistance**.



Σχήμα 6.5: Προτεινόμενη κλάση καθορισμού τιμών γνωρίσματος

Η κλάση **Distance** έχει τις υποκλάσεις που φαίνονται στο σχήμα. Στις περισσότερες περιπτώσεις που η κλάση αυτή γίνεται τιμή γνωρίσματος κατηγορίας, και θέλουμε να δημιουργήσουμε ένα γνώρισμα σαν περίπτωση αυτής της κατηγορίας, συμβαίνει να επιλέγουμε την υποκλάση της **MetricDistance** για κλάση καθορισμού τιμών γνωρίσματος. Ετσι συνδέουμε την **Distance** και την **MetricDistance**, με γνώρισμα κατηγορίας *default_ef*, όπως φαίνεται στο σχήμα.

Ο χρήστης μπορεί, αν θέλει, να αγνοήσει το προτεινόμενο ή τα προτεινόμενα πεδία τιμών του μοντέλου και να επιλέξει το επιθυμητό από όλες τις υποκλάσεις της τιμής γνωρίσματος της κατηγορίας. Αυτό γίνεται, όπως έχουμε ήδη αναφέρει, με κατάλληλη επιλογή από το ποντίκι (δες ενότητα 6.2.1).

6.3 Επέκταση πεδίου τιμών γνωρίσματος

Στις κάρτες ενημέρωσης γνωρισμάτων ο χρήστης έχει τη δυνατότητα να δημιουργήσει γνωρίσματα που έχουν για τιμή στοιχεία πρωτογενών τύπων, κόμβους που ήδη υπάρχουν στη βάση ή κόμβους που δημιουργούνται επί τούτου. Αυτή η τελευταία περίπτωση αποτελεί διευκόλυνση προς το χρήστη, ώστε να μη χρειάζεται να δημιουργεί τον κόμβο που θα αποτελέσει τιμή γνωρίσματος, με την

ειδική λειτουργία δημιουργίας κόμβου. Είναι πιο πρακτικό, όταν υπάρχει μια τέτοια απαίτηση, να δημιουργείται ο κόμβος και ταυτόχρονα[2] να συνδέεται με την ΕΟ, με το κατάλληλο γνώρισμα. Αυτό βέβαια οδηγεί σε επέκταση του πεδίου τιμών γνωρίσματος.

Δεν είναι όμως πάντα επιθυμητό να δημιουργούνται κόμβοι από την κάρτα ενημέρωσης γνωρισμάτων. Κάνοντας χρήση των οντοτήτων του μοντέλου περιορισμών και καθοδήγησης, αποφασίζεται αν πρέπει να δίνεται αυτή η δυνατότητα στο χρήστη. Η απόφαση λαμβάνεται για κάθε ενημερώσιμη κατηγορία ξεχωριστά ενώ τα σημεία που ελέγχονται είναι η κατηγορία του γνωρίσματος που πρόκειται να δημιουργηθεί, και η τιμή γνωρίσματος της κατηγορίας. Στη συνέχεια περιγράφεται η μεθοδολογία που έχουμε αναπτύξει για τη λήψη αυτής της απόφασης.

1. Αν η τιμή γνωρίσματος της κατηγορίας είναι μία από τις κλάσεις του συστήματος, τότε θα είναι είτε πρωτογενής τύπος, είτε η κλάση **Individual** :

   - αν είναι πρωτογενής τύπος (μία από τις κλάσεις **Telos_string**, **Telos_Integer**, **Telos_Real**), τότε τιμή γνωρίσματος μπορεί να είναι ένα στοιχείο του εκάστοτε τύπου. Σε αυτή την περίπτωση δεν δημιουργείται νέα τιμή γνωρίσματος, απλώς αναφέρεται μία υπάρχουσα[3], άρα δεν επεκτείνεται το πεδίο τιμών γνωρίσματος.
   - αν είναι η κλάση **Individual**, τότε
     -- αν ο χρήστης πληκτρολογήσει για τιμή γνωρίσματος ένα κόμβο που υπάρχει ήδη στη βάση, δεν επεκτείνεται το πεδίο τιμών γνωρίσματος.
     -- αν ο χρήστης πληκτρολογήσει για τιμή γνωρίσματος έναν κόμβο που δεν υπάρχει, αυτό σημαίνει πρόθεσή του να δημιουργήσει έναν καινούργιο. Καλείται να πληκτρολογήσει **και** το πεδίο τιμών γνωρίσματος, ή τουλάχιστο το επίπεδο ταξινόμησής του. Σε αυτή την περίπτωση επεκτείνεται το πεδίο τιμών γνωρίσματος.

2. Αν η τιμή γνωρίσματος της κατηγορίας είναι περίπτωση της κλάσης **RestrictedVocabulary**, τότε επιτρέπεται να δημιουργηθούν γνωρίσματα που έχουν μόνο ήδη υπάρχοντες κόμβους για τιμές γνωρισμάτων (δες ενότητα 5.2). Σε αυτή την περίπτωση δεν επεκτείνεται το πεδίο τιμών γνωρίσματος.

3. Αν η κατηγορία γνωρισμάτων είναι περίπτωση της μετα-κατηγορίας (**Individual**, *non_shared*) τότε τα γνωρίσματα που πρόκειται να δημιουργηθούν, μπορεί να παίρνουν τιμή γνωρίσματος μόνο κόμβους που δημιουργούνται αποκλειστικά γι' αυτό το σκοπό[4]. Σε αυτή την περίπτωση το πεδίο τιμών γνωρίσματος επεκτείνεται.

[2] Η δημιουργία του γνωρίσματος και του κόμβου που αποτελεί την τιμή γνωρίσματός του, γίνεται σε μία δοσοληψία.
[3] Σύμφωνα με το μοντέλο δεδομένων της Telos θεωρούμε ότι κάθε περίπτωση πρωτογενούς τύπου προϋπάρχει.
[4] Εαν υπάρχουν περιπτώσεις της τιμής γνωρίσματος της κατηγορίας που **δεν** αναφέρονται ως τιμή κάποιου γνωρίσματος της συγκεκριμένης κατηγορίας, τότε κατ' εξαίρεση προσφέρονται και αυτές οι περιπτώσεις στο χρήστη, για επιλογή.

4. Επιτρέπεται να δημιουργηθούν γνωρίσματα που έχουν τιμή γνωρίσματος είτε καινούργιους κόμβους είτε ήδη υπάρχοντες (που ταξινομούνται σαν περιπτώσεις του πεδίου τιμών ή επιλέγονται από αυτές, αντίστοιχα). Σε αυτή την περίπτωση μπορεί, κατ' επιλογή του χρήστη, να επεκταθεί το πεδίο τιμών γνωρίσματος.

Κεφάλαιο 7

Εμπειρία χρήσης

Η προσέγγιση που χρησιμοποιεί το ΔΕΔ για την ενημέρωση της πληροφορίας, είναι η διεργασιοκεντρική, όπως είδαμε και στο 2ο κεφάλαιο. Επομένως είναι λογικό να περιμένουμε να έχει επιτυχία, σε εφαρμογές που η εισαγωγή δεδομένων γίνεται κυρίως με αυτή τη μέθοδο. Ο έλεγχος που επιβάλλει το μοντέλο διεργασιών σε αρκετές περιπτώσεις εφαρμογών είναι επιθυμητός, ενώ σε άλλες μάλλον προβληματίζει το χρήστη. Οι ακολουθίες λειτουργιών και γενικότερα η έννοια της καθοδήγησης στη διαδικασία ενημέρωσης που προτείνεται από το μοντέλο, αν και βοηθά ιδιαίτερα σε μερικές περιπτώσεις, είναι ελάχιστα χρήσιμη σε άλλες. Σε αυτό το κεφάλαιο παραθέτουμε ένα σύνολο εφαρμογών στις οποίες το ΔΕΔ χρησιμοποιήθηκε. Εξετάζουμε τις ιδιαιτερότητες κάθε περίπτωσης και αναλύουμε τους λόγους που θεωρούμε ότι συνετέλεσαν στην επιτυχή ή όχι χρήση του ΔΕΔ σε αυτές.

7.1 Εφαρμογή SIB Class Management System

Η εφαρμογή **SIB Class Management System** [CDV93] του ΣΣΕ έχει σκοπό την τεκμηρίωση και ανάλυση στατικών ιδιοτήτων μονάδων λογισμικού. Παρέχει τη δυνατότητα περιγραφής και ανάκλησης μονάδων λογισμικού με βάση τη λειτουργία τους, με σκοπό την αναχρησιμοποίησή τους. Για να εξαχθούν οι στατικές ιδιότητες των μονάδων λογισμικού, χρησιμοποιούνται ειδικευμένοι μεταφραστές (parsers), που μεταφράζουν, στο μοντέλο δεδομένων της γλώσσας Telos, προγράμματα και βιβλιοθήκες λογισμικού, γραμμένα σε γλώσσες προγραμματισμού, όπως η C++ στην προκειμένη περίπτωση.

Με το ΔΕΔ, οι μονάδες λογισμικού ταξινομούνται διαλογικά από τον χρήστη με βάση την λειτουργία τους [CDa]. Η ταξινόμηση αυτή ονομάζεται *πολυεδρική ταξινόμηση* και αποτελείται από τέσσερεις ιεραρχίες εξειδίκευσης ή αλλιώς *όψεις ταξινόμησης* (*classification facets*): *Abstraction*, *Operation*, *OperatesOn* και *Dependencies* [Rub91].

Οι απαιτήσεις διαλογικής ενημέρωσης της εφαρμογής, καλύπτονται με τρεις διεργασίες

ενημέρωσης:

1. **Classify Software Object**: Επιτρέπει την *πολυεδρική ταξινόμηση* των μονάδων λογισμικού (*classes, member_functions, friend, operators, public*) που είναι δηλωμένες σε ένα αρχείο, το οποίο δίνεται σαν όρισμα στη διεργασία.

2. **Edit Term Hierarchy**: Επιτρέπει την εξέλιξη των *όψεων ταξινόμησης* λογισμικού, ενώ δέχεται σαν παράμετρο μια από τις τέσσερεις όψεις ταξινόμησης που θέλουμε να εξελιχθεί.

3. **Describe Library**: Παρέχει στο χρήστη τη δυνατότητα να προσθέσει και να περιγράψει νέες βιβλιοθήκες λογισμικού, οι οποίες παριστάνονται στο μοντέλο εφαρμογής.

Για λόγους που αυτή η εφαρμογή επέβαλε, προστέθηκε στο μοντέλο καθοδήγησης η κατηγορία *non_shared*. Σκοπό έχει την αποτροπή απόδοσης κοινού συνωνύμου σε περισσότερους του ενός όρων, όπως είδαμε και στο παράδειγμα της ενότητας 5.3.

Το μοντέλο της εφαρμογής περιγράφει ένα σχετικά απλό κόσμο, αυτόν της περιγραφής στατικών ιδιοτήτων μονάδων λογισμικού. Οι ενημερώσεις που είχαμε την απαίτηση να γίνουν σε αυτό είναι προβλέψιμες, και μπορούν να ελεγχθούν από ένα μοντέλο διεργασιών. Η επιβολή περιορισμών στις ενημερώσεις, με σκοπό να μην αλλοιωθούν τμήματα του μοντέλου της εφαρμογής, έγινε εύκολα με τη χρήση των παραπάνω διεργασιών. Το ΔΕΔ αρχικά σχεδιάστηκε για τις ανάγκες αυτής της εφαρμογής και τα αποτελέσματα που έδωσε η χρησιμοποίησή του ήταν ικανοποιητικά, όπως αναφέρει η σχετική αξιολόγηση του συστήματος από τη SIEMENS [Mei94].

7.2 Εφαρμογή ΚΛΕΙΩ

Το ΚΛΕΙΩ [Χρι94a] είναι ένα πληροφοριακό σύστημα μουσείων το οποίο μπορεί να χρησιμοποιηθεί σαν επιστημονικός κατάλογος μουσειακών αντικειμένων και καταχωρεί φυσικές, μορφολογικές, χρηστικές, ιστορικές, γεωγραφικές και χρονικές πληροφορίες. Το χαρακτηριστικό του είναι ότι διαθέτει ένα αρκετά σύνθετο μοντέλο, με δυνατότητα να εμπλουτίζεται διαρκώς. Οι σχέσεις μεταξύ των οντοτήτων της εφαρμογής μπορούν να είναι πολλαπλές και σύνθετες. Σε επίπεδο σχήματος, έχουν δημιουργηθεί ιεραρχίες εξειδίκευσης, μεγάλου βάθους, οι οποίες έχουν τη δυνατότητα να διευρύνονται.

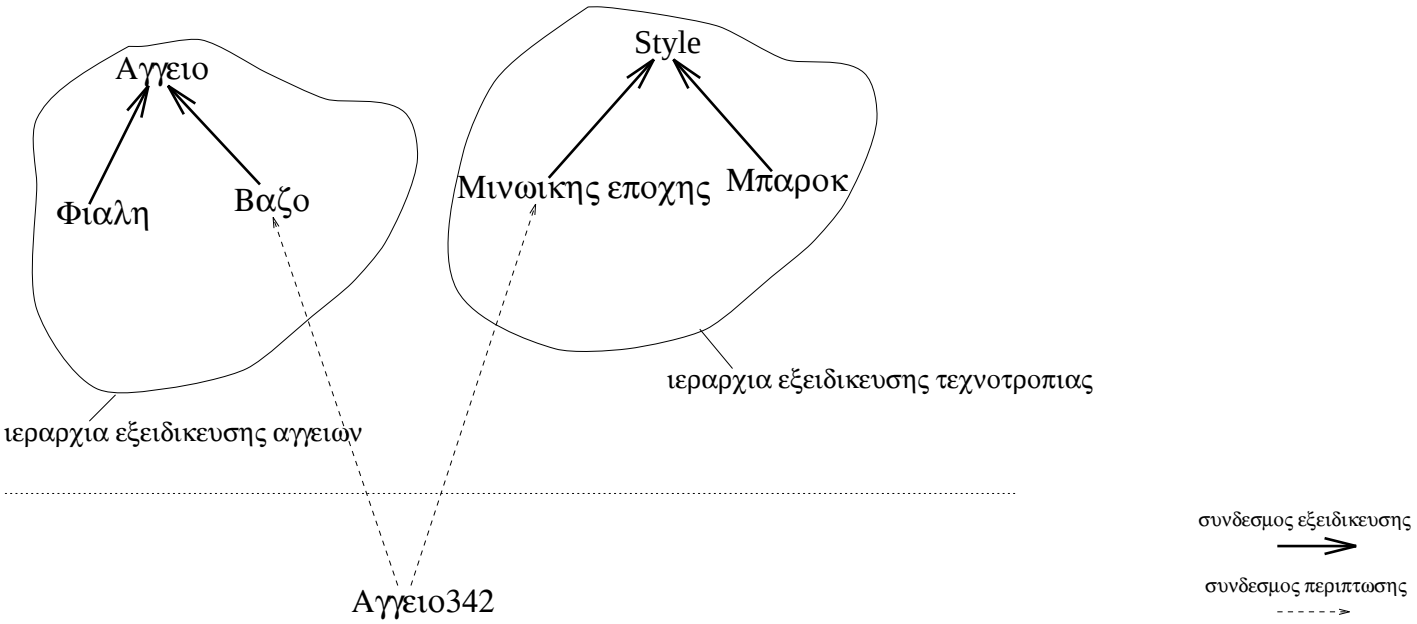
Οι απαιτήσεις διαλογικής ενημέρωσης της εφαρμογής καλύπτονται κυρίως με δύο γενικές διεργασίες ενημέρωσης[1]:

1. **Describe Class**: Επιτρέπει την εξέλιξη ιεραρχιών εξειδίκευσης, σε επίπεδο σχήματος. Παράμετρος σε αυτή τη διεργασία δίνεται μια κλάση, της οποίας την υπο-ιεραρχία εξειδίκευσης θέλουμε να εξελίξουμε.

[1]Εχουν χρησιμοποιηθεί κι άλλες διεργασίες, σε ρόλο βοηθητικό, όπως η **DescribeUnclassified**, που είδαμε στην ενότητα 3.4.3.

2. **Describe Object**: Επιτρέπει την εξέλιξη του πληθυσμού ιεραρχιών εξειδίκευσης. Παράμετρος σε αυτή τη διεργασία δίνεται μια κλάση, κορυφή της υπο-ιεραρχίας εξειδίκευσης, της οποίας τον πληθυσμό θέλουμε να εξελίξουμε.

Στο ΚΛΕΙΩ, είναι συχνή η απόδοση ιδιοτήτων σε μια οντότητα, με την ένταξή της σε περισσότερες από μια ιεραρχίες εξειδίκευσης. Για παράδειγμα, ένα συγκεκριμένο αγγείο, που είναι ενταγμένο στην υπο-ιεραρχία των αγγείων, μπορεί να ενταχθεί και στην υπο-ιεραρχία της τεχνοτροπίας (δες σχήμα 7.1). Οι δυο ιεραρχίες μπορεί να είναι ασύνδετες μεταξύ τους.



Σχήμα 7.1: Ενταξη οντότητας σε πολλαπλές ιεραρχίες εξειδίκευσης

Το **Αγγείο342** είναι ενταγμένο στις -ασύνδετες μεταξύ τους- ιεραρχίες εξειδίκευσης, που ορίζονται από τις κλάσεις **Αγγείο** και **Style**.

Για τον παραπάνω λόγο, η χρήση των γενικών παραμετρικών διεργασιών που αναφέραμε, ειδικά της **Describe Object**, δεν κρίθηκε ιδιαίτερα επιτυχής. Κι αυτό γιατί δεν επιτρέπεται σε μια οντότητα, που βρίσκεται σε μια ιεραρχία εξειδίκευσης, να ενταχθεί και σε μια άλλη ιεραρχία, όταν δεν έχουν κοινή υπερκλάση που να τις ενώνει (όπως στο σχήμα 7.1). Η εξέλιξη της γνώσης γίνεται στα πλαίσια δεδομένης ιεραρχίας.

Τελικά για να επιτευχθεί η ταξινόμηση μιας οντότητας σε κλάση όποιας ιεραρχίας εξειδίκευσης επιθυμούμε, ορίστηκε στη διεργασία **Describe Object** μια λειτουργία ενημέρωσης, η **Give Aspect**, με τύπο EDIT_CLASSES. Αυτή επιτρέπει την ταξινόμηση της οντότητας σε όποια κλάση, **κορυφή** ιεραρχίας εξειδίκευσης επιθυμούμε. Στη συνέχεια μπορούμε να ενημερώσουμε την οντότητα, σαν μέλος όμως *μίας συγκεκριμένης* υπο-ιεραρχίας εξειδίκευσης. Στην πράξη, για το παράδειγμά μας, αυτό σημαίνει ότι, για να εντάξουμε ένα δεδομένο αγγείο στην ιεραρχία της τεχνοτροπίας, πρέπει:

1. από τη διεργασία **Describe Object**, που περιγράφει αγγεία, να επιλέξουμε το ζητούμενο αγγείο

και να το ταξινομήσουμε (με τη λειτουργία **Give Aspect**) στην κορυφή της ιεραρχίας των τεχνοτροπιών, την κλάση **Style**.

2. από τη διεργασία **Describe Object**, που περιγράφει τεχνοτροπίες, να επιλέξουμε το ίδιο αγγείο και να το ταξινομήσουμε στην κατάλληλη τεχνοτροπία.

Η εναλλαγή των διεργασιών είναι κάτι που δυσκολεύει το χρήστη, και τελικά τον αναγκάζει για μια φαινομενικά απλή ενημέρωση -την ταξινόμηση μιας οντότητας- να κάνει περιττές κινήσεις.

Μια λύση που θα μπορούσε να δοθεί είναι η χρήση της μετα-κατηγορίας (**Individual**, *related_classes*), που προτείνεται και στο [Τζί95b]. Αυτή θα μπορούσε να συνδέει σχετικές μεταξύ τους ιεραρχίες (όπως αυτή των **Αγγείων**, με την αντίστοιχη των **Style**). Ετσι μια οντότητα ταξινομημένη σε μια ιεραρχία εξειδίκευσης, θα μπορούσε να ταξινομηθεί και σε όλες τις σχετικές με αυτήν ιεραρχίες, χωρίς εναλλαγή διεργασίας. Με αυτό τον τρόπο θα εξαλειφόταν η άσκοπη εναλλαγή καρτών, απαραίτητη για την αλλαγή διεργασίας, που όμως κουράζει το χρήστη. Θα μπορούσε ακόμα η ίδια η εναλλαγή διεργασίας να γίνει με πιο εύκολο τρόπο, όπως αναφέρουμε στο 8ο κεφάλαιο.

Σε αυτήν την εφαρμογή επίσης, παρουσιάστηκαν δυσκολίες στην ενημέρωση σύνθετων οντοτήτων. Οπως είδαμε και στην ενότητα 6.1, αυτό αντιμετωπίστηκε με την εισαγωγή της κατηγορίας (**Individual**, *dependent*) και τη χρήση των αναδρομικών καρτών ενημέρωσης γνωρισμάτων. Ακόμα, για τις ανάγκες του ΚΛΕΙΩ και για την παράσταση κλάσεων αναλλοίωτου πληθυσμού, προστέθηκε στο μοντέλο περιορισμών η μετα-κλάση **RestrictedVocabulary**, όπως είδαμε στην ενότητα 5.2.

Παρόλες όμως τις βελτιώσεις και τροποποιήσεις του ΔΕΔ για αυτήν την εφαρμογή, φαίνεται πως οι απαιτήσεις της ταιριάζουν στο είδος της ενημέρωσης που χαρακτηρίσαμε οντοκεντρική. Κάθε οντότητα είναι επιθυμητό να ενημερώνεται με όλες τις λειτουργίες που της επιτρέπονται. Επιπλέον ο χρήστης θέλει να έχει ελευθερία πλοήγησης στη βάση, κάτι που δεν μπορούν να του το προσφέρουν οι προκαθορισμένες από το μοντέλο ακολουθίες κινήσεων. Είναι μια επιλογή που κάνει ο ίδιος κι όχι ένα εκ των προτέρων καθορισμένο μοντέλο.

Βέβαια, παραμένουν διεργασίες, όπως η **Describe Unclassified**, που είδαμε στην ενότητα 3.4.3, στις οποίες είναι χρήσιμο να υπάρχει καθοδήγηση στην ενημέρωση, με ειδικευμένες λειτουργίες (ταξινόμηση σε ιεραρχίες εξειδίκευσης), που ενεργούν σε προκαθορισμένο πλήθος οντοτήτων (οι αταξινόμητες οντότητες της βάσης). Αυτές απευθύνονται σε χρήστες, που εξελίσσουν τη γνώση του μοντέλου της εφαρμογής, με σαφώς καθορισμένες, εκ των προτέρων, λειτουργίες, σε ελεγχόμενα τμήματά του.

Θα μπορούσαν να χρησιμοποιηθούν κι άλλες ειδικευμένες διεργασίες, που να αντικαθιστούν ή να συμπληρώνουν τις γενικές **Describe Class** και **Describe Object**. Ομως, η εμπειρία μας από την χρήση του ΔΕΔ στο ΚΛΕΙΩ, δείχνει πως, γενικά, η οντοκεντρική προσέγγιση είναι πιο ενδεδειγμένη για τις ενημερώσεις αυτής της εφαρμογής.

7.3 Εφαρμογή Booch

Σκοπός της εφαρμογής είναι να περιγράψει το περιβάλλον ανάπτυξης εφαρμογών και να δώσει στο χρήστη τη δυνατότητα να αποθηκεύσει, διαχειριστεί και εξαγάγει πληροφορία για συγκεκριμένες περιγραφές τμημάτων λογισμικού. Εχει σα βάση τη μεθοδολογία οντοκεντρικής ανάλυσης και σχεδίασης του Booch [Boo91]. Εχοντας μοντελοποιήσει και αποθηκεύσει γνώση για ένα περιβάλλον ανάπτυξης λογισμικού σε μια βάση δεδομένων, στόχος είναι η επίτευξη αναχρησιμοποίησης τμημάτων του.

Προσαρμόζοντας κατάλληλα τις γενικές διεργασίες ενημέρωσης του ΚΛΕΙΩ, κατασκευάστηκαν οι διεργασίες της εφαρμογής Booch. Αυτή τη φορά το σχήμα της βάσης δεν είναι πολύπλοκο, οι κλάσεις σε επίπεδο σχήματος είναι λίγες ενώ το βάθος των ιεραρχιών εξειδίκευσης μικρό. Για τις γενικές διεργασίες τα αποτελέσματα ήταν ικανοποιητικά. Το ΔΕΔ αξιολογήθηκε σαν ένα εύχρηστο και εύκολα προσαρμόσιμο εργαλείο για διαλογική ενημέρωση πληροφορίας, που δεν απαιτεί ειδική γνώση για τη χρησιμοποίησή του, από μέρους του χρήστη, σύμφωνα με τα [CM94a], [CM94b].

7.4 Εφαρμογή AAT

Σκοπός της εφαρμογής είναι η ανάπτυξη ενός θησαυρού όρων που περιέχει, εκτός από τους όρους αυτούς καθ' αυτούς, τις μεταξύ τους σχέσεις, έννοιες με τις οποίες κάθε όρος σχετίζεται και προτάσεις ανάπτυξης του θησαυρού από ειδικούς [GET95]. Οι όροι του AAT έχουν συγκεντρωθεί από ήδη υπάρχοντες καταλόγους, λεξικά, εγκυκλοπαίδειες, γλωσσάρια και επιστημονικές μονογραφίες. Περιέχει συνώνυμα όρων, ευρύτερους και πιο περιοριστικούς όρους, ιεραρχίες ταξινόμησης και αλφαβητικό ευρετήριο. Το ΣΣΕ επιδιώκει να γίνει το σύστημα συντονισμού του θησαυρού (Vocabulary Coordination System), βοηθώντας αρχικά στην παράσταση του θησαυρού όρων και στη συνέχεια στην εξέλιξή του.

Μέχρι τώρα το ΔΕΔ χρησιμοποιήθηκε δοκιμαστικά σε αυτή την εφαρμογή, με χρήση μιας διεργασίας, της **Describe Term**, που δίνει τη δυνατότητα για ενημέρωση όρων. Η διεργασία δίνει τη δυνατότητα για δημιουργία, διαγραφή, μετονομασία και ενημέρωση γνωρισμάτων στους όρους του θησαυρού.

Για τις ανάγκες αυτής της εφαρμογής, χρησιμοποιήθηκαν κατηγορίες γνωρισμάτων όπως η *interconnected* και η *necessary*, όπως είδαμε σε παραδείγματα στις ενότητες 5.1.2 και 5.4 αντίστοιχα. Οι περιορισμοί που επιβάλλουν αυτές οι μετα-κατηγορίες, κάνουν το ΔΕΔ ένα εργαλείο που εκτός από τη δυνατότητα εισαγωγής δεδομένων, προφυλάσσει την εννοιολογική ορθότητα της βάσης σε ικανοποιητικό βαθμό.

7.5 Γενικές παρατηρήσεις

Από τη χρήση του ΔΕΔ στις παραπάνω εφαρμογές, διαπιστώνουμε ότι το εργαλείο έχει χρησιμότητα στις εφαρμογές εκείνες για τις οποίες η διεργασιοκεντρική προσέγγιση στην ενημέρωση της πληροφορίας είναι δυνατή. Αν μπορούμε να απομονώσουμε τμήματα της βάσης που μπορούν να ενημερώνονται ανεξάρτητα από άλλα, τότε μπορούν να κατασκευαστούν εξειδικευμένες διεργασίες για την ενημέρωσή τους. Τέτοιες διεργασίες καθοδηγούν το χρήστη και συγχρόνως τον περιορίζουν σε ενημερώσεις της αρμοδιότητάς του. Σε αυτές τις περιπτώσεις το εργαλείο αποδεικνύεται φιλικό και εύχρηστο. Αν δεν είναι εφικτή η απομόνωση τμήματων της βάσης, για την κατασκευή εξειδικευμένων διεργασιών, λόγω της πολυπλοκότητας της εφαρμογής για παράδειγμα, τότε καταφεύγουμε στη χρήση γενικών διεργασιών, που ελπίζουμε ότι θα καλύψουν σύνολο ενημερώσεων. Ομως στην τελευταία περίπτωση τα αποτελέσματα δεν είναι εγγυημένα, όπως διαπιστώσαμε από τις εφαρμογές ΚΛΕΙΩ και Booch.

Ενδέχεται σε κάθε εφαρμογή να υπάρχει απαίτηση για εξειδικευμένες διεργασίες, όμως δεν μπορούμε να ισχυριστούμε ότι όλες οι ενημερώσεις μιας εφαρμογής είναι δυνατό να ενταχθούν στα πλαίσια μιας σαφώς καθορισμένης διεργασίας ενημέρωσης. Ετσι είδαμε ότι στις εφαρμογές ΚΛΕΙΩ και Booch, για να καλυφθούν ενημερώσεις σε μεγάλο σύνολο οντοτήτων, ορίστηκαν γενικές παραμετρικές διεργασίες. Η πολυπλοκότητα του μοντέλου στο ΚΛΕΙΩ, έκανε αυτές τις διεργασίες να μην αποδίδουν ικανοποιητικά, ενώ στο Booch, εφαρμογή με λιγότερο σύνθετο μοντέλο, τα σχόλια ήταν θετικά.

Ενα άλλο χαρακτηριστικό του ΔΕΔ είναι η δυνατότητα καθορισμού του πλήθους και της σειράς εκτέλεσης λειτουργιών που ανήκουν σε μία δοσοληψία. Από το μοντέλο διεργασιών καθορίζονται ακολουθίες λειτουργιών που λαμβάνουν μέρος σε μία δοσοληψία. Αν δεν έχει οριστεί ακολουθία λειτουργιών, τότε το ΔΕΔ ενημερώνει τη βάση με δοσοληψία που αποτελείται από μια μόνο λειτουργία κάθε φορά. Αυτό έχει την ιδιαιτερότητα ότι κάθε λειτουργία που επιλέγεται από το χρήστη έχει προκαθορισμένη εξέλιξη: είτε θα ενημερώσει τη βάση σε μία δοσοληψία μόνη της (αν δεν ανήκει σε ακολουθία λειτουργιών), είτε σε συνδυασμό με τις λειτουργίες της ακολουθίας της. Ικανοποιούνται έτσι διεργασίες εφαρμογών με απαίτηση για αυστηρή καθοδήγηση στη διαδικασία ενημέρωσης, όπως είδαμε στην ενότητα 5.5.

Από την άλλη, ενημερώσεις που απαιτούν ελευθερία κινήσεων δεν ικανοποιούνται από το μοντέλο διεργασιών, άρα ούτε από το ΔΕΔ. Αυτές ταιριάζουν περισσότερο στην οντοκεντρική προσέγγιση ενημέρωσης πληροφορίας. Στο [Τζί95b] παρουσιάζονται τα πλεονεκτήματα μιας τέτοιας προσέγγισης, κάνοντας χρήση ενός Μέτα Μοντέλου Οψεων Ενημέρωσης (ΜΜΟΕ). Το τελευταίο δίνει τη δυνατότητα για έλεγχο της ενημέρωσης ανά οντότητα, αφού για κάθε μία μπορούν να οριστούν επιτρεπτές λειτουργίες. Επίσης επιτρέπει την ελεύθερη πλοήγηση στη βάση, αφού δεν περιορίζει το χρήστη στην ενημέρωση προκαθορισμένων οντοτήτων.

Ιδανικά και οι δύο προσεγγίσεις, διεργασιοκεντρική και οντοκεντρική, χρειάζονται και μάλιστα συνδυαζόμενες. Το ΜΜΟΕ (οντοκεντρική προσέγγιση) μπορεί να χρησιμοποιηθεί για τον έλεγχο των ενημερώσεων ανά οντότητα. Το ΜΔΕ (διεργασιοκεντρική προσέγγιση) μπορεί να χρησιμοποιηθεί για τον έλεγχο των ενημερώσεων ανά διεργασία και τον καθορισμό ακολουθιών από λειτουργίες ενημέρωσης για αυτήν. Ο συνδυασμός των δύο μοντέλων μπορεί να οδηγήσει σε διαλογική ενημέρωση που από τη μια να επιτρέπει ελεύθερη πλοήγηση και ενημέρωση της βάσης, ενώ από την άλλη να επιτρέπει την επιλογή και εκτέλεση διεργασιών ενημέρωσης, με ακολουθίες προκαθορισμένων λειτουργιών σε ελεγχόμενα τμήματα της βάσης.

Κεφάλαιο 8

Επίλογος

Στο τελευταίο κεφάλαιο παραθέτουμε τα συμπεράσματα της εργασίας και μερικές βελτιώσεις και επεκτάσεις που μπορούν να γίνουν στο μοντέλο διεργασιών, στο μοντέλο περιορισμών και στην επαφή χρήσης του ΔΕΔ.

8.1 Συμπεράσματα

Στην παρούσα εργασία ασχοληθήκαμε με το πρόβλημα της διαλογικής ενημέρωσης οντοκεντρικών βάσεων δεδομένων. Χρησιμοποιήθηκε ένα μοντέλο διεργασιών[1] που βασίζεται στη διεργασιοκεντρική προσέγγιση, για την περιγραφή της διαλογικής ενημέρωσης. Με βάση αυτό το μοντέλο, ο χρήστης, έχοντας σαν κέντρο της ενημέρωσης μια διεργασία, μπορεί να επιλέξει οντότητες ‘‘σχετικές’’ με αυτήν, και να τις επηρεάσει με ‘‘σχετικές’’ ενημερώσεις.

Στηριζόμενοι σε αυτό το μοντέλο, σχεδιάσαμε και κατασκευάσαμε το Δελτίο Εισαγωγής Δεδομένων. Με βάση τις προδιαγραφές που δίνονται από το μοντέλο διεργασιών, το εργαλείο ρυθμίζει κατάλληλα τη λειτουργικότητά του και προσαρμόζεται στις ανάγκες κάθε εφαρμογής που καλείται να ενημερώσει. Το ΔΕΔ μπορεί να διακρίνει τους χρήστες και να τους επιτρέπει ενημερώσεις, μέσω διεργασιών, που έχουν καθοριστεί στο εν λόγω μοντέλο.

Το ΔΕΔ προσφέρει ένα πλήρες σύνολο από πράξεις για ενημέρωση της βάσης, αφού υποστηρίζει όλες τις στοιχειώδεις ενημερώσεις που μπορεί να εφαρμοστούν σε οντότητες των οντοκεντρικών βάσεων του ΣΣΕ. Οι πράξεις αυτές υλοποιούνται μέσω καρτών ενημέρωσης που επιτρέπουν δημιουργία, διαγραφή, μετονομασία, ταξινόμηση, γενίκευση, και απόδοση γνωρισμάτων για κόμβους και γνωρίσματα.

Το εργαλείο έχει τη δυνατότητα να διατηρεί ένα σύνολο από περιορισμούς ακεραιότητας, με τη χρήση ενός μοντέλου περιορισμών που αναπτύχθηκε για αυτό το σκοπό. Ετσι διατηρούνται περιορισμοί που δεν ελέγχονται από το σύστημα σημασιολογικού ελέγχου της Telos, όπως

[1]Το μοντέλο διεργασιών κατασκευάστηκε από την Ομάδα Πληροφοριακών Συστημάτων και Τεχνολογίας Λογισμικού του ΙΠ-ΙΤΕ. Η εξειδίκευσή του για τις εφαρμογές στις οποίες χρησιμοποιήθηκε, έγινε κυρίως από το Γιάννη Τζίτζικα.

διατήρηση σύνδεσης ιεραρχιών εξειδίκευσης, διατήρηση συνόλων αναλλοίωτου πληθυσμού, αποφυγή απόδοσης κοινών τιμών σε γνωρίσματα για τα οποία κάτι τέτοιο επιβάλλεται, διατήρηση γνωρισμάτων απαραίτητων για την ύπαρξη κόμβων.

Εχει αναπτυχθεί ένα σύνολο αυτοματισμών, με σκοπό τη διευκόλυνση του χρήστη κατά τη διάρκεια της διαλογικής ενημέρωσης. Ετσι, ο χρήστης μπορεί, κατά τη δημιουργία γνωρισμάτων, να αποδίδει τιμές σε αυτά, επιλέγοντάς τις από το επιθυμητό πεδίο τιμών. Επίσης από το σύστημα προτείνεται όνομα για τις εξαρτημένες οντότητες, ενώ ο χρήστης απλά επικυρώνει την επιλογή αυτή, αν είναι επιθυμητή. Τέλος, διευκολύνεται η ενημέρωση σύνθετων οντοτήτων, με δυνατότητα ενημέρωσης των οντοτήτων που τις απαρτίζουν.

Το ΔΕΔ χρησιμοποιήθηκε, σαν εργαλείο του ΣΣΕ, για τη διαλογική ενημέρωση βάσεων που στηρίζονται στο μοντέλο δεδομένων της γλώσσας παράστασης γνώσης Telos. Χρησιμοποιήθηκε και αξιολογήθηκε στις εφαρμογές SIB Class Management System, AAT, Booch και ΚΛΕΙΩ. Στις πρώτες τρεις είχε αρκετή επιτυχία, ενώ στην τελευταία η χρήση του δεν έδωσε τα αναμενόμενα αποτελέσματα. Αυτό οφείλεται στο γεγονός ότι οι χρήστες μιας τέτοιας εφαρμογής, προτιμούν κατά κύριο λόγο την ενημέρωση της βάσης με την οντοκεντρική προσέγγιση[2].

Τελικά μπορούμε να πούμε ότι το ΔΕΔ είναι κατάλληλο να χρησιμοποιηθεί σαν εργαλείο για καθοδηγούμενη διαλογική ενημέρωση των οντοκεντρικών βάσεων του ΣΣΕ. Προσφέροντας τη δυνατότητα για καθορισμό ακολουθιών από λειτουργίες, ικανοποιούνται εφαρμογές που έχουν απαίτηση για αυστηρή καθοδήγηση στη διαδικασία ενημέρωσης. Ο έλεγχος των ενημερώσεων ανά διεργασία, προσέγγιση την οποία το εργαλείο χρησιμοποιεί, δίνει τη δυνατότητα για περιορισμό των ενημερώσεων στις απόλυτα επιτρεπτές και επιθυμητές. Αποδίδοντας τις διεργασίες σε χρήστες του συστήματος, δίνεται τελικά η δυνατότητα για έλεγχο των ενημερώσεων ανά χρήστη.

8.2 Επεκτάσεις-Βελτιώσεις

8.2.1 Μοντέλο διεργασιών

Στο ΜΔΕ, όπως είδαμε, μπορεί να αποδοθεί σε μια λειτουργία ένα σύνολο ορισμάτων, οπότε η λειτουργία περιορίζεται υποχρεωτικά από το όρισμα αυτό. Αν δεν υπάρχει το όρισμα, τότε επιτρέπονται στη λειτουργία όλες οι σημασιολογικά ορθές ενημερώσεις που αντιστοιχούν στον τύπο της. Θα μπορούσε όμως το σύνολο ορισμάτων να χαρακτηριστεί **προτεινόμενο** για τη λειτουργία, από το μοντέλο διεργασιών, οπότε μπορεί να ερμηνευτεί ως εξής κατά περίπτωση:

- Αν είναι λειτουργία τύπου CREATE_NODE το σύνολο περιέχει τις πιθανές κλάσεις του προς δημιουργία κόμβου. Αυτές παρουσιάζονται στο χρήστη, ενώ ο τελευταίος μπορεί να τις

[2]Αντίθετα με τη διεργασιοκεντρική προσέγγιση, που έχει σαν κέντρο της ενημέρωσης μια διεργασία, η οντοκεντρική έχει σαν κέντρο της ενημέρωσης μια οντότητα, προσφέροντας λειτουργίες για την ενημέρωσή της.

αποδεχτεί, να αφαιρέσει κάποια ή να προσθέσει κάποια που αυτός θέλει (αρκεί να υπάρχει στη βάση).

- Αν είναι λειτουργία τύπου EDIT_CLASSES το σύνολο περιέχει ένα προτεινόμενο σύνολο από κλάσεις για την εστιακή οντότητα, που μπορεί να παρουσιάζονται, όπως τώρα, σε λίστα για επιλογή από το χρήστη. Παρόλα αυτά η εστιακή οντότητα μπορεί να ταξινομηθεί και σε κάποια από τις κλάσεις που θα πληκτρολογήσει ο χρήστης - ακόμα κι αν αυτή δεν υπάρχει στο σύνολο ορισμάτων. Ακόμα μπορεί να διαγράψει συνδέσμους περίπτωσης από τις κλάσεις που είναι τώρα ταξινομημένο, χωρίς να περιορίζεται από το σύνολο ορισμάτων.

- Αν είναι λειτουργία τύπου EDIT_SUPERCLASSES ισχύουν ανάλογοι κανόνες με την περίπτωση λειτουργίας τύπου EDIT_CLASSES μόνο που αντί για συνδέσμους περίπτωσης, ενημερώνονται σύνδεσμοι εξειδίκευσης.

- Αν είναι λειτουργία τύπου UPDATE_ATTRIBUTES, το σύνολο περιέχει τις προτεινόμενες κατηγορίες γνωρισμάτων, κάτω από τις οποίες επιτρέπεται να δημιουργήσουμε ή να διαγράψουμε γνωρίσματα. Με μια επιλογή που θα προσφέρεται, ο χρήστης μπορεί να αγνοήσει αυτές τις κατηγορίες και να ζητήσει όλες τις πιθανές κατηγορίες γνωρισμάτων για την εστιακή οντότητα.

- Αν είναι λειτουργία τύπου DELETE_NODE ή RENAME_NODE ο χαρακτηρισμός του συνόλου ως προτεινόμενου είναι μάλλον τυπικός, αφού για τη λειτουργία θα δηλώνει ό,τι και τώρα δηλώνει η απουσία συνόλου ορισμάτων: πλήρης ελευθερία στις αντίστοιχες ενημερώσεις.

Το μοντέλο διεργασιών που καθορίζει τη λειτουργικότητα του ΔΕΔ, μπορεί να γίνει πιο εκφραστικό, με την παράσταση λειτουργιών ενημέρωσης για γνωρίσματα. Ετσι θα γίνει αποδοτικότερη η ενημέρωσή τους, θέτοντας περιορισμούς στις ενημερώσεις που ο χρήστης μπορεί να εφαρμόσει σε αυτά. Εξάλλου, με αυτό τον τρόπο θα γίνει ισότιμη μεταχείριση λειτουργιών για κόμβους και για γνωρίσματα.

Μια ιδιαίτερα σημαντική επέκταση θα ήταν η χρησιμοποίηση ακολουθιών από λειτουργίες ενημέρωσης, οι οποίες να μην δρουν αναγκαστικά στην ίδια εστιακή οντότητα (όπως ισχύει τώρα), αλλά να επιτρέπεται ο καθορισμός ακολουθιών από λειτουργίες με διαφορετικές εστιακές οντότητες. Ετσι θα υπήρχε η έννοια της προκαθορισμένης από το μοντέλο δοσοληψίας, που θα βοηθούσε στον καθορισμό σύνθετων λειτουργιών. Βέβαια και τώρα υπάρχει η έννοια της προκαθορισμένης δοσοληψίας, αλλά σε περιορισμένη μορφή, αφού οι λειτουργίες της δρουν στην ίδια εστιακή οντότητα.

Στο μοντέλο περιορισμών του ΔΕΔ, θα μπορούσε να προστεθεί η μετα-κατηγορία (**Individual**, *related_classes*), όπως αναφέραμε και στην ενότητα 7.2. Η χρησιμότητά της είναι ιδιαίτερα μεγάλη

σε μοντέλα που χρησιμοποιούν τη σχέση πολλαπλής ταξινόμησης, για απόδοση χαρακτηριστικών ιδιοτήτων, όπως είδαμε ότι συμβαίνει στο ΚΛΕΙΩ.

Σημαντική βελτίωση στην εξέλιξη της γνώσης και του ελέγχου της ενημέρωσης μέσα από το ΔΕΔ, θα ήταν η χρησιμοποίηση του Μετα Μοντέλου Οψεων Ενημέρωσης (ΜΜΟΕ), που προτείνεται από τον Τζίτζικα στο [Τζί95b]. Το μοντέλο διεργασιών, καθοδήγησης και περιορισμών που χρησιμοποιεί το ΔΕΔ, μπορεί να ενοποιηθεί με το εν λόγω μετα-μοντέλο, με σκοπό την αποτελεσματικότερη εξέλιξη της γνώσης. Το ΜΜΟΕ κάνει χρήση θετικών και αρνητικών δηλώσεων ενημέρωσης, επιτρέποντας ή αποτρέποντας ενημερώσεις για δεδομένες οντότητες. Βασικά πλεονεκτήματά του είναι η παροχή ελευθερίας πλοήγησης και ενημέρωσης της βάσης καθώς και ο έλεγχος των ενημερώσεων ανά οντότητα. Αντιστοιχεί στην οντοκεντρική προσέγγιση ενημέρωσης της πληροφορίας.

Ιδανικά οι δύο προσεγγίσεις, αυτή του ΜΔΕ και αυτή του ΜΜΟΕ, μπορούν να συνδυαστούν, οδηγώντας σε ένα εργαλείο ενημέρωσης δεδομένων που από τη μια να επιτρέπει ελεύθερη πλοήγηση και ενημέρωση της βάσης, ενώ από την άλλη να επιτρέπει την επιλογή και εκτέλεση διεργασιών ενημέρωσης, με λειτουργίες που ανήκουν σε προκαθορισμένες ακολουθίες.

8.2.2 Επαφή χρήσης

8.2.2.1 Κάρτα επιλογής διεργασίας

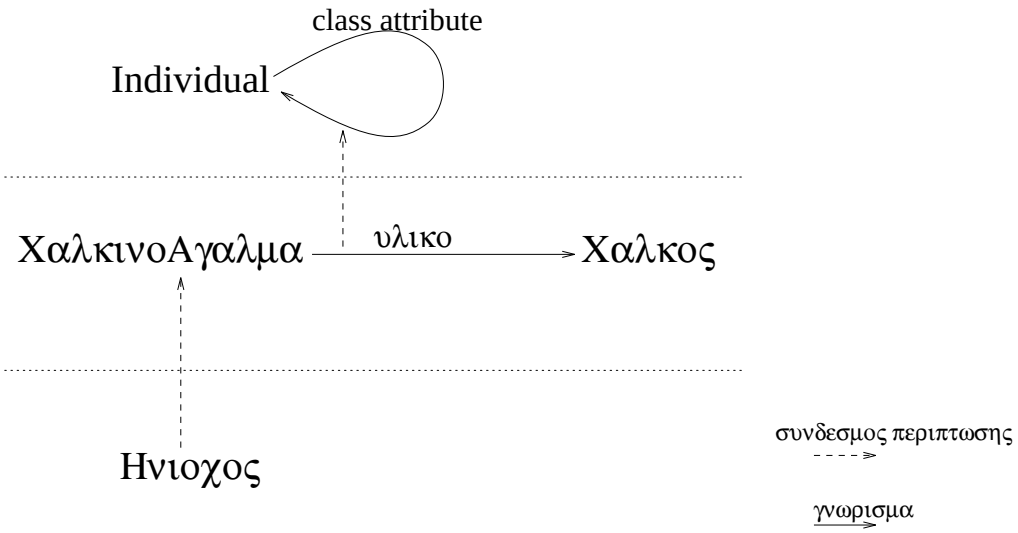
Μια επιθυμητή βελτίωση της επαφής χρήσης του ΔΕΔ, θα ήταν η τροποποίηση της επιλογής διεργασίας. Θα μπορούσε αυτή η κάρτα και η κάρτα επιλογής λειτουργίας για κόμβους (η πρώτη και δεύτερη κάρτα του ΔΕΔ αντίστοιχα) να συγχωνευτούν σε μία. Οι λειτουργίες των δύο αυτών καρτών μπορούν να ενσωματωθούν, οπότε έχουμε το πλεονέκτημα ότι επιταχύνεται η αλλαγή διεργασίας και ο επαναπροσδιορισμός ορίσματος στην ίδια διεργασία (πράξεις αρκετά χρήσιμες, όπως φάνηκε από την μέχρι τώρα εμπειρία, ειδικά στην εφαρμογή του ΔΕΔ στο ΚΛΕΙΩ).

8.2.2.2 Εισαγωγή/διαγραφή χαρακτηριστικών γνωρισμάτων κλάσεων

Από το μεταφραστή της Telos έχει δοθεί η δυνατότητα για εισαγωγή χαρακτηριστικών γνωρισμάτων σε κλάσεις (class attributes). Αυτά ουσιαστικά είναι γνωρίσματα που έχει η κλάση αλλά τα κληρονομούν όλες οι περιπτώσεις της. Είναι γνωρίσματα που φαίνεται να έχει κάθε περίπτωση της κλάσης, αλλά που δεν μπορεί να τα αλλάξει, αφού αναφέρονται σε ολόκληρη την κλάση.

Εστω για παράδειγμα η κλάση **ΧάλκινοΑγαλμα**. Αυτή έχει χαρακτηριστικό γνώρισμά της το *υλικό*, με τιμή γνωρίσματος το **Χαλκός**. Μια περίπτωση της κλάσης αυτής είναι ο **Ηνίοχος**, που έχει γνώρισμα με τιμή το **Χαλκός**, αφού το κληρονομεί από την κλάση του (δες σχήμα 8.1).

Το ΔΕΔ πρέπει να δίνει τη δυνατότητα στην κλάση **ΧάλκινοΑγαλμα** να μπορεί να αλλάξει το χαρακτηριστικό της γνώρισμα, αν αυτό είναι επιθυμητό, όμως ο κόμβος **Ηνίοχος** να μην μπορεί, αφού το γνώρισμα **Χαλκός** έχει οντότητα εκκίνησης την ίδια την κλάση (όπως φαίνεται και στο



Σχήμα 8.1: Χαρακτηριστικά γνωρίσματα κλάσεων

Η κλάση **ΧάλκινοΑγαλμα** έχει χαρακτηριστικό γνώρισμά της το *υλικό*, με τιμή γνωρίσματος το **Χαλκός**, το οποίο κληρονομεί ο **Ηνίοχος**

σχήμα 8.1). Εξάλλου αλλαγή του χαρακτηριστικού γνωρίσματος της κλάσης από μια περίπτωσή της, θα είχε επιρροή και σε άλλες περιπτώσεις της κλάσης, κάτι που δεν είναι επιθυμητό. Πρέπει λοιπόν να εξασφαλίσουμε ειδική μεταχείριση γι' αυτά τα ειδικά γνωρίσματα.

Προτείνεται η δημιουργία ειδικής κάρτας λειτουργίας που θα ασχολείται με την εισαγωγή/διαγραφή τέτοιων γνωρισμάτων από τις κλάσεις που τα έχουν ως χαρακτηριστικά. Πρέπει επίσης όλες οι περιπτώσεις της κλάσης να παρουσιάζουν απλώς αυτά τα γνωρίσματα, χωρίς να προσφέρουν τη δυνατότητα αλλαγής τους (κατάσταση <readonly>).

8.2.3 Ερωτηματική γλώσσα

Τέλος, αναφέρουμε μια βελτίωση που αναμένεται να γίνει στη χρήση της ερωτηματικής γλώσσας του ΣΣΕ και θα επηρεάσει θετικά το μέχρι τώρα δύσχρηστο τρόπο καθορισμού συνόλων.

Οπως είδαμε στην ενότητα 3.3.1, ο καθορισμός συνόλων στο μοντέλο διεργασιών γίνεται με τη βοήθεια QueryMacro, και χρήση της ερωτηματικής γλώσσας του ΣΣΕ, το qi. Ομως η χρήση της γλώσσας αυτής είναι αρκετά επίπονη, για το μη ειδικευμένο χρήστη, αφού δεν είναι ιδιαίτερα φιλική. Υπάρχει δυσκολία κατανόησης του συνόλου που το QueryMacro δίνει σαν απάντηση. Εξάλλου η χρήση λογικών ονομάτων στον κώδικα καθορισμού ενός QueryMacro, μπορεί να προκαλέσει ασυνέπεια, όταν η οντότητα με το δεδομένο λογικό όνομα ενημερωθεί από μια λειτουργία.

Η χρήση της TELQUEL, της νέας ερωτηματικής γλώσσας του ΣΣΕ που σχεδιάζεται, υπόσχεται να λύσει τα παραπάνω προβλήματα. Θα είναι πιο φιλική στο χρήστη, με συντακτικό παρόμοιο της γνωστής SQL. Ακόμα, όταν μια οντότητα με δεδομένο λογικό όνομα αναφέρεται από μια δομή αντίστοιχη του QueryMacro, γραμμένη στην TELQUEL, θα απαγορεύεται από το σύστημα

σημασιολογικού ελέγχου η ενημέρωση της. Ετσι θα εξαλειφθούν πιθανές ασυνέπειες που έχουμε στην παρούσα έκδοση, με την τροποποίηση οντοτήτων που παίρνουν μέρος σε QueryMacro.

Παράρτημα Α

Περιγραφή μοντέλου διεργασιών σε Telos

```
TELL Individual Users in S_Class with
    attribute
        task        : UpdateTask
end


TELL Individual UpdateTask in S_Class with
    attribute
        object      : QueryMacro;
        export      : QueryMacro ;
        update      : Update
end


TELL Individual Update in S_Class with
    attribute
        update_type: UpdateType;
        argument   : QueryMacro;
        sequence   : Update
end


TELL Individual UpdateType in S_Class
end
```

```
TELL Individual CREATE_NODE in Token ,
                                UpdateType
end


TELL Individual DELETE_NODE in Token ,
                                UpdateType
end


TELL Individual RENAME_NODE in Token ,
                                UpdateType
end


TELL Individual EDIT_CLASSES in Token ,
                                UpdateType
end


TELL Individual EDIT_SUPERCLASSES in Token ,
                                UpdateType
end


TELL Individual UPDATE_ATTRIBUTES in Token ,
                                UpdateType
end
```

Βιβλιογραφία

[Πατ93a] Ελένη Πατάκη. ‘‘Σχεδίαση και Υλοποίηση ενός Εργαλείου Επιλογής Λογισμικού για Αναχρησιμοποίηση”. Μεταπτυχιακή εργασία, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης, Απρίλιος 1993.

[Ντα93b] Κώστας Νταντουρής. ‘‘Βιβλιοθήκη στοιχειωδών ερωτηματικών συναρτήσεων και επεξεργασία ερωτήσεων στη γλωσσα TELOS”. Μεταπτυχιακή εργασία, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης, 1993.

[Χρι94a] Μαρία Χριστοφοράκη. ‘‘Τεκμηρίωση πολιτιστικών Αγαθών με το σύστημα ΚΛΕΙΩ”. Μεταπτυχιακή εργασία, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης, Φεβρουάριος 1994.

[Γεω94b] Γιώργος Γεωργιαννάκης. ‘‘Ο Μηχανισμός Αποθήκευσης και Διαχείρισης Οντοτήτων για τη Γλώσσα Παράστασης Γνώσης Telos”. Μεταπτυχιακή εργασία, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης, Φεβρουάριος 1994.

[Θεο95a] Μάνος Θεοδωράκης. ‘‘Εμβέλεια Ονόματος σε Σημασιολογικά Μοντέλα Δεδομένων”. Μεταπτυχιακή εργασία, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης, Σεπτέμβριος 1995.

[Τζί95b] Γιάννης Τζίτζικας. ‘‘Έλεγχος και Διευκόλυνση της Εξέλιξης Γνώσης στην γλώσσα TELOS”. Μεταπτυχιακή εργασία, Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης, Οκτώβριος 1995.

[Ban92] Francois Bancilhon. ‘‘The O_2 Object-Oriented Database System”. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, volume 21, San Diego, California, June 1992.

[BDD+92] Theodosia Bitzou, Dimitris Daskalakis, Martin Dörr, Polivios Klimathianakis, Nikos Prekas, and Maria Theodoridou. ‘‘SIB Static Analyser : Runtime System”. Technical report, Information Systems and Software Techology Group, Institute of Computer Science Foundation of Research and Technology-Hellas, December 1992.

[Boo91] G. Booch. ‘‘*Object-Oriented Design with Applications*”. Benjamin/Cummings Publishing Company, 1991.

[BOS91] Paul Butterworth, Allen Otis, and Jacob Stein. ‘‘The GemStone Object Database Management System”. *Communications of the ACM*, 34(10):64--77, October 1991.

[CDa] Panos Constantopoulos and Martin Dörr. ‘‘Component Classification in the Software Information Base”. in O.Nierstrasz and D.Tsichritzis, eds.,Object-Oriented Software Composition, Prentice-Hall, 1995.

[CDb] Panos Constantopoulos and Martin Dörr. ‘‘The Semantic Intex System - A Brief Presentation”. Information Systems and Software Technology Group, Institute of Computer Science Foundation of Research and Technology Hellas.

[CDP+92] Panos Constantopoulos, Martin Dörr, Elena Pataki, Eleni Petra, Maria Theodoridou, and Yannis Vassiliou. ‘‘Design of the Software Information System”. Technical Report ITHACA.FORTH.92E2#1 TR.E.3, Information Systems and Software Techology Group, Institute of Computer Science Foundation of Research and Technology Hellas, January 1992.

[CDV93] Panos Constantopoulos, Martin Dörr, and Yannis Vassiliou. ‘‘Repositories for Software Reuse : The Software Information Base”. In *Proceedings IFIP WG 8.1 Conference on Information System Development Process*, pages 285--307, Como,Italy, September 1993.

[CM94a] Yannis Charalabidis and Nano Mourtzinou. ‘‘Object-Oriented Design and Implementation Management and Reuse on the SIS Plaform”. Technical Report Ithaca.CMSU.94.MOD#3, CMSU, July 1994.

[CM94b] Yannis Charalabidis and Nano Mourtzinou. ‘‘SIB for Design and Implementation Reuse (SIB/DI) Users Manual”. Technical Report Ithaca.CMSU.94.MOD#5, CMSU, December 1994.

[CP] Panos Constantopoulos and Elena Pataki. ‘‘A Browser for Software Reuse”. In *Proceedings 4th International Conference on Advanced Information Systems Engineering - CAiSE 92, Manchester, May 1992.*

[DD94] Costas Dadouris and Martin Dörr. ‘‘The Query Interface for SIS”. Technical Report 0, Information Systems and Software Techology Group, Institute of Computer Science Foundation of Research and Technology Hellas, February 1994.

[DDP93] Costas Dadouris, Martin Dörr, and Nikos Prekas. ‘‘Answerer: An Interactive Program to Use PQI functions”. Technical Report 4, Information Systems and Software Techology Group, Institute of Computer Science Foundation of Research and Technology Hellas, October 1993.

[DDPM93] Costas Dadouris, Martin Dörr, Elena Pataki, and Dimitris Mauroidis. ‘‘Implementation of the SIB System”. Technical Report ITHACA.FORTH.92E2#7 TR.E.3, Information Systems and Software Techology Group, Institute of Computer Science Foundation of Research and Technology Hellas, January 1993.

[Deux90] O.Deux et al. ‘‘The Story of O_2”. *IEEE Transactions on Knowledge and Data Engineering*, 2(1):91--107, March 1990.

[Deux91] O.Deux et al. ‘‘The O_2 System”. *Communications of the ACM*, 34(10):34--48, October 1991.

[Dix93] A. Dix et al. ‘‘*Human-Computer Interaction*”. Prentice Hall, 1993.

[DKT93] Martin Dörr, Polivios Klimathianakis, and Manos Theodorakis. ‘‘SIS Data Entry Language User’s Manual”. Technical Report Version 1.1, Information Systems and Software Techology Group, Institute of Computer Science Foundation of Research and Technology Hellas, October 1993.

[DT94] Dimitris Daskalakis and Yannis T. Tzitzikas. ‘‘Customizing Data Entry Forms”. Technical Report 9, Information Systems and Software Techology Group, Information Systems and Software Technology Group Hellas, October 1994.

[EM93] Henrik Eriksson and Mark Musen. ‘‘Metatools for Knowledge Acquisition”. *IEEE Software*, 10(3):23--29, May 1993.

[Eri] Henrik Eriksson. ‘‘Specification and Generation of Custom-Tailored Knowledge-Acquisition Tools”. http://www-ksl.stanford.edu/KSL_Abstracts/KSL-93-50.html.

[GET95] ‘‘Vocabulary Coordination System - Request for comment”, March 1995. The Getty Art History Information Program.

[Kim90] Won Kim. ‘‘*Introduction to Object-Oriented Databases*”. MIT Press, 1990.

[Mad90] F. Maddix. ‘‘*Human Computer Interaction. Theory and Practice*”. Ellis Horwood, 1990.

[May92] D. Mayhew. ‘‘*Principles and Guidelines in Software User Interface Design*”. Prentice Hall, 1992.

[MBJK90] John Mylopoulos, Alex Borgida, Matthias Jarke, and Manolis Koubarakis. ‘‘Telos: Representing Knowledge about Information Systems”. *ACM Transactions on Information Systems*, 8(4), October 1990.

[Mei94] J. Meier. ‘‘Evaluation report (Summary): SIB Static Analysis and Class Management System V1.2”, October 1994. Siemens Nixdorf.

[Mus89] Mark Musen. ‘‘Knowledge Acquisition at the Metalevel: Creation of Custom-Tailored Knowledge-Acquisition Tools”. *SIGART Newsletter*, (108):45--55, April 1989.

[MWKE93] Katharina Morik, Stefan Wrobl, Jong-Uwe Kietz, and Werner Emde. ‘‘*Knowledge Acquisition and Machine Learning*”. Academic Press, 1993.

[Nor] Donald Norman. in Ronald Baeker and William Buxton, eds., Readings in Human-Computer Interaction, (Los Altos, Calif.: Morgan Kaufmann Publishers, Inc, 1987).

[PEEM] Angel Puerta, Henrik Eriksson, John Egar, and Mark Musen. ‘‘Generation of Knowledge-Acquisition Tools from Reusable Domain Ontologies”. http://www-ksl.stanford.edu/KSL_Abstracts/KSL-92-81.html.

[PM88] Joan Peckham and Fred Maryanski. ‘‘Semantic Data Models”. *ACM Computing Surveys*, 20(3):153--189, September 1988.

[Rod90] Ulka Rodgers. ‘‘*Unix Database Management Systems*”. Yourdon Press Computing Series, 1990.

[Rub91] Prieto-Diaz Ruben. ‘‘Implementing Faceted Classification for Software Reuse”. *Communications of the ACM*, 34(5), 1991.

[Shn92] B. Shneiderman. ‘‘*Designing the User Interface. Strategies for Effective Human-Computer Interaction*”. Addison-Wesley, 1992.

[Sol92] Valery Soloviev. ‘‘An Overview of Three Commercial Object-Oriented Database Management Systems: ONTOS, ObjectStore, and O_2”. *SIGMOD RECORD*, 21(1):93--104, March 1992.

[Ull89] Jeffrey Ullman. ‘‘*Principles of Database and Knowledge Base Systems*”, volume 2. Computer Science Press, 1989.

[ZCC95] Mansour Zand, Val Collins, and Dale Caviness. ‘‘A Survey of Current Object-Oriented Databases”. *Data Base Advances*, 26(1):14--29, February 1995.