ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΚΑΤΑΙΟΣ ΙΙ:
Λογισμικό Σύστημα για την Αναπαράσταση και Εξέλιξη Βάσεων Δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΦΩΤΕΙΝΗΣ Π. ΑΝΑΓΝΩΣΤΟΥ

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2007

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

ΕΚΑΤΑΙΟΣ ΙΙ:
Λογισμικό Σύστημα για την Αναπαράσταση και Εξέλιξη Βάσεων Δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΦΩΤΕΙΝΗΣ Π. ΑΝΑΓΝΩΣΤΟΥ

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 19η Νοεμβρίου 2007.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Ιωάννης Βασιλείου	Τιμολέων Σελλής	Κωνσταντίνος Κοντογιάννης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αναπληρωτής Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2007

(Υπογραφή)

.....

ΦΩΤΕΙΝΗ Π. ΑΝΑΓΝΩΣΤΟΥ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Ο σκοπός της διπλωματικής εργασίας ήταν η ανάπτυξη ενός λογισμικού συστήματος, του ΕΚΑΤΑΙΟΥ ΙΙ, που θα επιτρέπει την διεξαγωγή υποθετικών σεναρίων όσον αφορά στην εξέλιξη του σχήματος μιας βάσης δεδομένων.

Το σύστημα αναπαριστά τμήματα του σχήματος μιας βάσης δεδομένων, καθώς και τα ερωτήματα και τις όψεις που είναι ορισμένα πάνω σ' αυτή με τη μορφή κατευθυνόμενου γράφου, πάνω στους κόμβους του οποίου μπορούν να οριστούν πιθανά γεγονότα, όπως, για παράδειγμα, «Διαγραφή γνωρίσματος» και πολιτικές χειρισμού αυτών των γεγονότων, όπως «Στην περίπτωση διαγραφής γνωρίσματος τότε εμπόδισε την αλλαγή».

Για κάθε γεγονός που θα συμβεί σε κάθε κόμβο μπορεί να οριστεί εξαρχής ποια πολιτική θα ακολουθηθεί. Όταν χρειαστεί να γίνει μια αλλαγή στη βάση, όπως διαγραφή ενός γνωρίσματος, ορίζεται ένα νέο γεγονός «διαγραφή γνωρίσματος» στον αντίστοιχο κόμβο. Στη συνέχεια τονίζεται ο γράφος του σχήματος της βάσης δεδομένων και των ερωτημάτων, με τρόπο που δείχνει πως επηρεάζεται κάθε στοιχείο του, λόγω της εισαγωγής του νέου γεγονότος, των υπαρχόντων πολιτικών και των αλληλεξαρτήσεων μεταξύ των στοιχείων του γράφου.

Ένα επιπλέον πλεονέκτημα που προσφέρει το σύστημα είναι η οπτικοποίηση του γράφου και ο γραφικός χειρισμός του. Έτσι, γίνεται περισσότερο εμφανής η επίδραση που επιφέρει μια αλλαγή στο σχήμα της βάσης δεδομένων στο σύστημα και ευκολότερη η απόφαση για την αποδοχή της ή μη.

Λέξεις Κλειδιά: <<Εξέλιξη σχήματος βάσεων δεδομένων, Αναπαράσταση σε γράφο>>

Abstract

The scope of this thesis is to develop a software system, called “Hecataeus II”, which allows the conduct of hypothetical scenarios, regarding the evolution of a database schema.

The system represents parts of a database schema along with the queries and views defined on it as a directional graph, on which nodes hypothetical scenarios can be assigned, such as “Delete attribute” and policies handling these scenarios, for instance “In case of attribute deletion then block the modification”.

The type of the policy that should be followed if an event occurs, could be defined from the very beginning. In case of a change in the database schema, such as an attribute deletion, a new event “Attribute deletion” is assigned on the respective node. Then, the graph is highlighted in a way that presents how each element is affected, given the new event definition, the existing policies and the interdependence among the graph elements.

An extra advantage that the system offers, is the visualization of the graph and its graphical handling. Thus, the impact of a change in the database schema is more obvious and the decision for its approval or not is easier.

Keywords: <<Database schema evolution, Graph representation>>

Πρόλογος

Φτάνοντας στο τέλος των προπτυχιακών μου σπουδών, θα ήθελα να ευχαριστήσω τους καθηγητές μου και ιδιαίτερα τον επιβλέποντα για τη διπλωματική μου εργασία κ. Ιωάννη Βασιλείου. Επίσης ευχαριστώ τον συνεπιβλέποντα Γιώργο Παπαστεφανάτο για την άψογη συνεργασία. Θα ήθελα να ευχαριστήσω και από αυτή τη θέση τους γονείς μου, Παναγιώτη και Δέσποινα, για την ευκαιρία που μου έδωσαν να σπουδάσω πάνω στο αντικείμενο που μου αρέσει και τη συμπαράστασή τους όλα αυτά τα χρόνια, καθώς και τον αδερφό μου, Γιάννη, ο οποίος με κάθε τρόπο στήριξε την προσπάθειά μου κατά τη συγγραφή της διπλωματικής. Τέλος, ευχαριστώ τους συμφοιτητές μου για τις όμορφες αναμνήσεις από τα χρόνια στη σχολή, τον Δημήτρη, την Πέρσα και την Πόπη.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 *Εισαγωγή*

Στην ενότητα αυτή περιγράφεται ο γενικότερος χώρος στον οποίο εμπίπτει η διπλωματική, η εξέλιξη του σχήματος των βάσεων δεδομένων, τα προβλήματα που ο χώρος αυτός αντιμετωπίζει και ο τρόπος με τον οποίο σκοπεύει να συνεισφέρει η παρούσα διπλωματική στην επίλυση αυτών. Συγκεκριμένα, στην παράγραφο 1.1 περιγράφεται ο τομέας της εξέλιξης του σχήματος των βάσεων δεδομένων, τα χαρακτηριστικά του και τα προβλήματα που αντιμετωπίζει. Στη συνέχεια, στην παράγραφο 1.2 περιγράφεται το αντικείμενο της διπλωματικής και η προσφορά της στον τομέα της εξέλιξης του σχήματος των βάσεων δεδομένων, όσον αφορά τα προβλήματα που αναλύθηκαν.

1.1 Εξέλιξη σχήματος βάσης δεδομένων

Μια βάση δεδομένων είναι ένα δυναμικό σύστημα με πολλές αλληλεξαρτήσεις μεταξύ των στοιχείων που την αποτελούν. Καθώς αλλάζουν οι ανάγκες και οι απαιτήσεις από μια βάση, είναι απαραίτητο να αλλάξει και το σχήμα της, ούτως ώστε να τις ικανοποιεί. Δυστυχώς, μια μικρή αλλαγή σε ένα στοιχείο του σχήματος της βάσης, όπως η διαγραφή ενός γνωρίσματος ή η μετονομασία μιας σχέσης, ενδέχεται να επηρεάσει άλλα στοιχεία του σχήματος της βάσης, ή ερωτήματα και όψεις που εξαρτώνται από αυτά.

Έστω, για παράδειγμα, μια βάση δεδομένων που περιλαμβάνει μια σχέση ΥΠΑΛΛΗΛΟΙ με γνωρίσματα ΟΝΟΜΑΤΕΠΩΝΥΜΟ, ΔΙΕΥΘΥΝΣΗ και ΤΗΛΕΦΩΝΟ. Έστω επίσης ότι έχει οριστεί ένα ερώτημα που εμφανίζει όλα τα στοιχεία ενός υπαλλήλου. Αν για κάποιους λόγους η εταιρία αποφασίσει ότι θέλει να αποθηκεύει στη βάση και το πατρώνυμο των υπαλλήλων, τότε απαιτείται η εισαγωγή ενός επιπλέον γνωρίσματος ΠΑΤΡΩΝΥΜΟ στη σχέση ΥΠΑΛΛΗΛΟΙ. Αυτή η αλλαγή επηρεάζει σαφώς το παραπάνω ερώτημα: είτε πρέπει να εμφανίζονται μόνο τα γνωρίσματα ΟΝΟΜΑΤΕΠΩΝΥΜΟ, ΔΙΕΥΘΥΝΣΗ και ΤΗΛΕΦΩΝΟ

και όχι όλα τα γνωρίσματα της σχέσης, είτε να εμφανίζεται επιπλέον και το γνώρισμα ΠΑΤΡΩΝΥΜΟ.

Συνεπώς, όταν απαιτείται να γίνει μια αλλαγή στη βάση δεδομένων, ο αναλυτής ή ο διαχειριστής της βάσης πρέπει να βρει ποια στοιχεία του σχήματος της βάσης, των ερωτημάτων και γενικά του συστήματος επηρεάζονται από αυτή την αλλαγή και να καθορίσει πως θα μετασχηματιστούν, ούτως ώστε να ενσωματώσουν ή όχι την αλλαγή. Η εργασία αυτή απαιτεί την αφιέρωση πολύ χρόνου και είναι επιρρεπής σε λάθη. Επιπλέον, το μέγεθος των βάσεων δεδομένων κάνει τη διαδικασία αυτή ακόμα δυσκολότερη.

1.2 Αντικείμενο διπλωματικής

Όπως αναφέρθηκε και στην προηγούμενη ενότητα, μια αλλαγή στο σχήμα της βάσης δεδομένων επηρεάζει συνήθως ένα μεγάλο τμήμα αυτής και των ερωτημάτων/όψεων που την προσπελαύνουν. Θα ήταν χρήσιμο στον διαχειριστή της βάσης αν ήξερε ποια στοιχεία του σχήματος της βάσης και των ερωτημάτων/όψεων επηρεάζονται από μια αλλαγή και ποιες ενέργειες θα πρέπει γίνουν για να μετασχηματιστεί η βάση, ούτως ώστε να ενσωματώσει την αλλαγή. Με αυτό τον τρόπο θα μπορούσε να αποφασίσει ευκολότερα αν επιθυμεί να αποδεχτεί την αλλαγή και αν ναι, να κάνει εφαρμόσει αυτές τις ενέργειες που θα μετασχηματίσουν σωστά τη βάση.

Αντικείμενο της παρούσας διπλωματικής είναι η δημιουργία ενός λογισμικού συστήματος που αναπαριστά τη βάση δεδομένων, καθώς και τις όψεις και τα ερωτήματα που την προσπελαύνουν με τη μορφή κατευθυνόμενου γράφου, ενώ επιτρέπει τη δημιουργία υποθετικών αλλαγών στο σχήμα της βάσης δεδομένων και την εύρεση των επιπτώσεών τους στο σύστημα. Το σύστημα επιτρέπει στο χρήστη να ορίσει τον τρόπο που θέλει να αντιμετωπιστεί ένα πιθανό γεγονός από τα διάφορα στοιχεία της βάσης και των ερωτημάτων με τον ορισμό πολιτικών. Για κάθε πιθανό γεγονός, ο χρήστης μπορεί να επιλέξει αν θέλει να εμποδιστεί ή να προωθηθεί η τροποποίηση που αυτό επιφέρει, ορίζοντας αντίστοιχες πολιτικές. Επίσης, μπορεί να ορίσει ένα γεγονός που επιθυμεί να εφαρμόσει στο σχήμα της βάσης και να βρει, δοθέντων των πολιτικών που υπάρχουν για αυτό το γεγονός και των αλληλεξαρτήσεων μεταξύ των στοιχείων του συστήματος, αν πρέπει να γίνουν τροποποιήσεις στο σύστημα ως αποτέλεσμα του γεγονότος και ποιες είναι αυτές.

Για καλύτερη εικόνα του γράφου του σχήματος της βάσης και των ερωτημάτων/όψεων που την προσπελαύνουν, καθώς και των τροποποιήσεων σε περίπτωση ορισμού γεγονότος, το σύστημα προσφέρει επιπλέον την οπτικοποίηση του γράφου και το γραφικό χειρισμό του.

1.2.1 Συνεισφορά

Συνεπώς, η συνεισφορά της διπλωματικής είναι ένα λογισμικό σύστημα που επιτρέπει:

1. Την αναπαράσταση του σχήματος της βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν με ενιαίο τρόπο ως κατευθυνόμενο γράφο.
2. Την ενσωμάτωση σημασιολογίας στο γράφο, όσον αφορά στην εξέλιξη του σχήματος της βάσης δεδομένων.
3. Τη δημιουργία υποθετικών αλλαγών και εύρεση επιπτώσεων.
4. Την οπτικοποίηση και το γραφικό χειρισμό του γράφου.

1.3 Οργάνωση κειμένου

Η δομή του κειμένου έχει ως εξής:

Η περιγραφή του θέματος της διπλωματικής γίνεται στο Κεφάλαιο 2. Παρουσιάζονται σχετικές εργασίες και το θεωρητικό υπόβαθρο για την αναπαράσταση και εξέλιξη του σχήματος της βάσης δεδομένων. Αρχικά παρουσιάζεται με συντομία ο τρόπος αναπαράστασης του σχήματος της βάσης δεδομένων και των ερωτημάτων προς αυτή με τη μορφή κατευθυνόμενου γράφου. Στη συνέχεια παρουσιάζεται αναλυτικά ο τρόπος εισαγωγής σημασιολογίας στο γράφο όσον αφορά στην εξέλιξη. Στο Κεφάλαιο 3 παρουσιάζονται οι απαιτήσεις από την αρχιτεκτονική του συστήματος και οι λειτουργίες που πρέπει να επιτελεί καθένα από τα βασικά τμήματα του συστήματος. Η λεπτομερής αρχιτεκτονική του συστήματος και οι λειτουργίες/μέθοδοι κάθε τμήματος παρουσιάζονται στο κεφάλαιο 4. Στο κεφάλαιο 5 παρουσιάζονται θέματα υλοποίησης, όπως οι πλατφόρμες και τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν στην υλοποίηση, καθώς και οι απαιτήσεις για την εγκατάσταση του ΕΚΑΤΑΙΟΥ ΙΙ σε έναν υπολογιστή. Στο κεφάλαιο 6 ελέγχεται η αξιοπιστία του συστήματος με τη βοήθεια ενός σεναρίου χρήσης. Το κεφάλαιο εξυπηρετεί και ως εγχειρίδιο χρήσης του προγράμματος. Στο κεφάλαιο 7 συνοψίζονται τα αποτελέσματα της διπλωματικής και αναφέρονται πιθανές μελλοντικές επεκτάσεις αυτής. Ακολουθεί το παράρτημα (κεφάλαιο 8). Τέλος, στο κεφάλαιο 9 παρουσιάζεται η σχετική βιβλιογραφία.

2 Σχετικές εργασίες και θεωρητικό υπόβαθρο

Η παρούσα διπλωματική έχει ως στόχο τη δημιουργία ενός λογισμικού συστήματος, του ΕΚΑΤΑΙΟΥ ΙΙ, το οποίο επιτρέπει την αναπαράσταση του σχήματος μιας βάσης δεδομένων, καθώς και των ερωτημάτων και των όψεων που την προσπελαύνουν με τη μορφή κατευθυνόμενου γράφου και τη μελέτη των επιπτώσεων της εξέλιξης του σχήματος της βάσης στην ίδια τη βάση και στα ερωτήματα/όψεις.

Σ' αυτό το κεφάλαιο περιγράφουμε εργασίες που σχετίζονται με το θέμα της διπλωματικής, καθώς και τα θέματα που πρέπει να έχει κατανοήσει ο αναγνώστης πριν από την παρουσίαση της ανάλυσης και σχεδίασης του συστήματος. Συγκεκριμένα, στην παράγραφο 2.1 παρουσιάζονται οι σχετικές με το θέμα εργασίες. Στην παράγραφο 2.2 περιγράφεται ο τρόπος αναπαράστασης της βάσης και των ερωτημάτων/όψεων που την προσπελαύνουν με τη μορφή κατευθυνόμενου γράφου. Τέλος, στην παράγραφο 2.3 περιγράφεται ο τρόπος εισαγωγής σημασιολογίας για εξέλιξη στα στοιχεία του γράφου και ο τρόπος εύρεσης της τροποποιημένης βάσης λόγω ενός γεγονότος.

Βάση αυτών παρουσιάζεται στην παράγραφο 2.4 ο σκοπός της διπλωματικής.

2.1 Σχετικές εργασίες

Ένας μεγάλος αριθμός από ερευνητικές εργασίες σχετίζεται με το πρόβλημα της εξέλιξης βάσεων δεδομένων. Στο [Rodd95] παρουσιάζεται μια ανασκόπηση για το πρόβλημα τήρησης εκδόσεων σχημάτων βάσεων δεδομένων και την εξέλιξή τους (schema versioning and evolution), ενώ στο [Rodd00] παρέχεται μια κατηγοριοποίηση όλων των ερευνητικών ζητημάτων σχετικά με την εξέλιξη βάσεων δεδομένων. Σύμφωνα με αυτήν την

κατηγοριοποίηση, μπορούμε να διακρίνουμε ερευνητικές εργασίες που αφορούν: (α) στην εξέλιξη αντικειμενοστραφών βάσεων δεδομένων [RaRu95], [Bane87], [Zica91], (β) στην εξέλιξη των μοντέλων οντοτήτων-συσχετίσεων, όπου οι αλλαγές θεωρούνται στο εννοιολογικό επίπεδο της βάσης [LiCC94], (γ) στην εξέλιξη στο λογικό επίπεδο της βάσης (εξέλιξη σχημάτων και τήρηση εκδόσεων) [Rodd95]), και τέλος (δ) στην εξέλιξη όψεων (view evolution), κυρίως στα πλαίσια των αποθηκών δεδομένων. Το πρόβλημα της εξέλιξης όψεων εξειδικεύεται στο πρόβλημα της προσαρμογής τους (view adaptation) στις ενδεχόμενες αλλαγές που συμβαίνουν στην βάση. Το πρόβλημα αυτό έχει 2 πλευρές: η πρώτη αφορά στην προσαρμογή υλοποιημένων όψεων (materialized view) σε αλλαγές που συμβαίνουν στον ορισμό της ίδιας της όψης, με στόχο την αποφυγή του επαναϋπολογισμού της όψης [GMRR01], [MoDo96], ενώ η δεύτερη αφορά στην προσαρμογή όψεων σε αλλαγές που συμβαίνουν στο σχήμα της βάσης [Bell02], [NiLR98], [RuLN97].

Σχετικό με το πρόβλημα της εξέλιξης βάσεων δεδομένων είναι το πρόβλημα της διαχείρισης μοντέλων (model management), το οποίο αποτελεί μια προσέγγιση στη διαχείριση των μετα-δεδομένων μιας βάσης. Με βάση τη διαχείριση μοντέλων, οι κύριες αφαιρετικές δομές είναι τα μοντέλα (π.χ. σχήματα βάσεων δεδομένων) και οι αντιστοιχήσεις (mappings) μεταξύ αυτών [Bern03], [Meln04], [VeMP04]. Το πρόβλημα της εξέλιξης σχημάτων βάσεων δεδομένων μπορεί λοιπόν να θεωρηθεί ως μια ειδική περίπτωση διαχείρισης μοντέλων, όπου αλλαγές στο σχήμα της βάσης θεωρούνται σαν αλλαγές στο μοντέλο που αντιστοιχεί στο σχήμα αυτό. Τα 2 μοντέλα, που αντιστοιχούν στα σχήματα πριν και μετά τις δομικές αλλαγές στη βάση, θα πρέπει να ανακτήσουν τις αντιστοιχήσεις μεταξύ τους. Δηλαδή, κατάλληλες αντιστοιχήσεις θα πρέπει να δημιουργηθούν μεταξύ των 2 αυτών μοντέλων, έτσι ώστε οι διάφορες επερωτήσεις και γενικότερα εφαρμογές που προσπελαύνουν τη βάση δεδομένων να προσαρμοστούν σε αυτές τις αλλαγές.

2.2 Αναπαράσταση σχήματος βάσης δεδομένων με γράφο.

Η αναπαράσταση του σχήματος της βάσης δεδομένων και των ερωτημάτων που την προσπελαύνουν με ενιαίο τρόπο, είχε μελετηθεί και παρουσιαστεί σε μια παρελθούσα εργασία [PKVV05], αντικείμενο της οποίας ήταν η δημιουργία ενός λογισμικού εργαλείου, του ΕΚΑΤΑΙΟΥ. Ο ΕΚΑΤΑΙΟΣ είναι η προηγούμενη έκδοση του ΕΚΑΤΑΙΟΥ ΙΙ και απεικονίζει με την μορφή ενός γράφου τα βασικά συστατικά ενός πληροφοριακού συστήματος και τα συστατικά εκείνα που αλληλεπιδρούν με την βάση δεδομένων, με φιλικό προς τον χρήστη τρόπο. Στις παραγράφους που ακολουθούν περιγράφεται το τμήμα της εργασίας που χρησιμοποιήθηκε για την παρούσα διπλωματική και το οποίο πρέπει να έχει διαβάσει ο αναγνώστης για να κατανοήσει το αντικείμενο της διπλωματικής.

2.2.1 Αναγκαιότητα αναπαράστασης του σχήματος βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν με γράφο.

Η αναπαράσταση του σχήματος της βάσης με τις παραδοσιακές τεχνικές μοντελοποίησης, όπως το διάγραμμα Οντοτήτων-Συσχετίσεων, τα διαγράμματα UML, κ.λ.π., παρόλο που είναι ευρέως χρησιμοποιούμενα και αναπαριστούν με παραστατικότητα τα στοιχεία του σχήματος μιας βάσης δεδομένων, όπως τις σχέσεις και τα γνωρίσματα, δεν αναπαριστούν άλλες έννοιες που εξαρτώνται από τη βάση δεδομένων, όπως τα ερωτήματα ή οι όψεις. Η αναπαράσταση του σχήματος των βάσεων δεδομένων και των ερωτημάτων/όψεων μπορεί να γίνει με ενιαίο τρόπο χρησιμοποιώντας τη μοντελοποίηση με γράφο. Επιπλέον, με την εν λόγω αναπαράσταση γίνονται εμφανέστερες οι αλληλεξαρτήσεις μεταξύ των στοιχείων του σχήματος της βάσης και των ερωτημάτων/όψεων και είναι ευκολότερη η μελέτη της εξέλιξης του σχήματος της βάσης.

2.2.2 Περιγραφή της τεχνικής μοντελοποίησης

Στην παράγραφο αυτή περιγράφεται η τεχνική μοντελοποίησης.

Ορισμός 1: Έστω G = (V, E) κατευθυνόμενος ακυκλικός γράφος, όπου V είναι ένα πεπερασμένο σύνολο κόμβων, και E είναι ένα πεπερασμένο σύνολο ακμών. Ο γράφος που θα κατασκευαστεί είναι ένας κατευθυνόμενος ακυκλικός γράφος όπου κάθε κόμβος v ανήκει σε ένα από τα ακόλουθα είδη κόμβων:

- *Κόμβοι Οντοτήτων*
- *Κόμβοι Γνωρισμάτων*
- *Κόμβοι Ερωτήσεων*
- *Κόμβοι Περιορισμών*
- *Κόμβοι Τελεστών*
- *Κόμβοι Σταθερών*
- *Κόμβοι Ομαδοποίησης Κατά*

Για την αναπαράσταση ενός πληροφοριακού συστήματος με την χρήση κατευθυνόμενου γράφου, η σημειολογία των χρησιμοποιούμενων κόμβων φαίνεται στον Πίνακα 2.1.

Σύμβολο	**Ορισμός**
`R, S, …`	κόμβος Οντότητας
`A1, A2, Bn, …`	κόμβος Γνωρίσματος
`Q, Q1, Q2, …`	κόμβος Ερώτησης
`R.PK`	κόμβος Περιορισμού Πρωτεύοντος Κλειδιού
`R.A1.FK`	κόμβος Περιορισμού Ξένου Κλειδιού

R.A1.UC	κόμβος Περιορισμού Μοναδικότητας
R.A1.CC	κόμβος Περιορισμού Ελέγχου
R.A1.NK	κόμβος Περιορισμού Μη Μηδενικής Τιμής
Op	κόμβος Δυαδικού Τελεστή
constant	κόμβος Σταθεράς (τιμή)
GB	κόμβος Ομαδοποίησης Κατά

Πίνακας 2.1. Σημειολογία Κόμβων

Το σύνολο Ε περιλαμβάνει τις προσανατολισμένες ακμές του προαναφερθέντος γράφου, και κάθε ακμή του συνόλου Ε ανήκει σε ένα από τα παρακάτω είδη ακμών:

- *Ακμή Σχήματος*: Αντιπροσωπεύει την συσχέτιση ανάμεσα σε μια οντότητα ή μια ερώτηση με το σχήμα της (π.χ. τα γνωρίσματα από τα οποία αποτελείται ένας πίνακας)
- *Ακμή Αντιστοίχησης*: Αντιπροσωπεύει την αντιστοίχηση του σχήματος οντοτήτων.
- *Ακμή Τελεστή*: Αντιπροσωπεύει τη συμμετοχή γνωρισμάτων σε περιορισμούς και πράξεις επιλογής ή συνένωσης.
- *Ακμή Όπου*: Αντιπροσωπεύει το τμήμα *Όπου* μίας ερώτησης SQL.
- *Ακμή Από*: Αντιπροσωπεύει το τμήμα *Από* μίας ερώτησης SQL.
- *Ακμή Ομαδοποίησης Κατά*: Αντιπροσωπεύει το τμήμα *Ομαδοποίηση Κατά* μίας ερώτησης SQL.
- *Ακμή ψευδώνυμου*: Αντιπροσωπεύει την σχέση ψευδωνύμου.

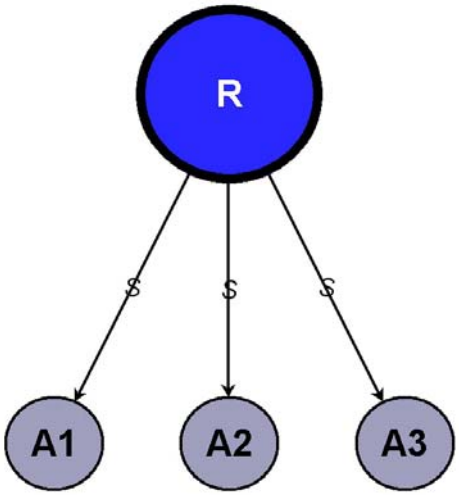
Η σημειολογία των χρησιμοποιούμενων ακμών φαίνεται στον Πίνακα 2.2.

Σύμβολο	**Ορισμός**
S	Ακμή που υποδεικνύει το σχήμα μίας οντότητας
op	Ακμή που υποδεικνύει τελεστή
Map-select	Ακμή που υποδεικνύει την αντιστοίχηση ανάμεσα στα γνωρίσματα μίας ερώτησης και τα γνωρίσματα μίας οντότητας
where	Ακμή που υποδεικνύει το τμήμα *Όπου* μιας ερώτησης
from	Ακμή που υποδεικνύει το τμήμα *Από* μιας ερώτησης
group by	Ακμή που υποδεικνύει το τμήμα *Ομαδοποίηση Κατά* μιας ερώτησης
alias	Ακμή που υποδεικνύει σχέση ψευδωνύμου

Πίνακας 2.2. Σημειολογία Ακμών

2.2.2.1 Οντότητες

Έστω R (A_1, A_2,…, A_n) μια οντότητα με γνωρίσματα A_1, A_2, …, A_n.Η αναπαράσταση με χρήση γράφου είναι ένας κατευθυνόμενος γράφος που έχει ακριβώς n+1 κόμβους και n ακμές. Ο γράφος είναι για την ακρίβεια ένα δέντρο που έχει σαν ρίζα τον κόμβο R και όλα τα γνωρίσματα σαν φύλλα. Οι ακμές είναι προσανατολισμένες και κατευθύνονται από την ρίζα προς τα φύλλα . Στην Εικόνα 2.1 παρουσιάζεται ένα παράδειγμα για την περίπτωση n=3.



Εικόνα 2.1. Αναπαράσταση Οντότητας

2.2.2.2 Περιορισμοί

Διακρίνουμε τους παρακάτω πέντε τύπους περιορισμών:

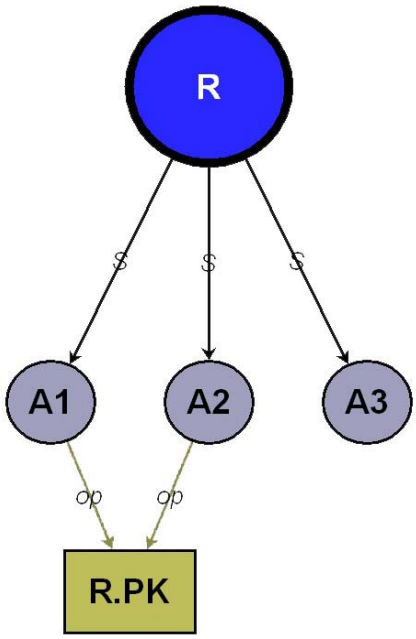
- *Περιορισμός Πρωτεύοντος Κλειδιού*
- *Περιορισμός Ξένου Κλειδιού*
- *Περιορισμός Μοναδικής Τιμής Γνωρίσματος*
- *Περιορισμός Μη Κενής Τιμής Γνωρίσματος*
- *Περιορισμός Ελέγχου Τιμής Γνωρίσματος*

2.2.2.2.1 Περιορισμός Πρωτεύοντος Κλειδιού

Έστω R (A_1, A_2,…, A_n) μια οντότητα με πρωτεύον κλειδί τα πεδία A_1, A_2, …, A_m. Η αναπαράσταση της οντότητας αποτελείται από έναν κατευθυνόμενο γράφο με *n+2* κόμβους και *n+m* ακμές. Οι *n+2* κόμβοι αντιστοιχούν σε έναν κόμβο που αναπαριστά την οντότητα, n κόμβους που αναπαριστούν τα γνωρίσματα και έναν κόμβο που αναπαριστά τον περιορισμό πρωτεύοντος κλειδιού. Επίσης έχουμε *n* ακμές τύπου *σχήματος* με ετικέτα *S* που ξεκινούν από τον κόμβο της οντότητας R και κατευθύνονται σε κάθε γνώρισμα. Επίσης έχουμε *m* ακμές τύπου *Περιορισμού* για τον περιορισμό πρωτεύοντος κλειδιού με ετικέτα *c* που ξεκινούν από τα γνωρίσματα που αποτελούν το πρωτεύον κλειδί και καταλήγουν στον κόμβο που

αναπαριστά τον περιορισμό. Η αναπαράσταση του Περιορισμού Πρωτεύοντος Κλειδιού είναι στην πραγματικότητα μια επέκταση της αναπαράστασης μίας οντότητας συνδεδεμένης με τον περιορισμό μέσω του αντίστοιχου κόμβου. Για την αναπαράσταση του περιορισμού χρησιμοποιούμε έναν κόμβο τύπου *Περιορισμού* με ετικέτα *<Όνομα_Οντότητας>.PK*.

Στην Εικόνα 2.2 παρουσιάζεται ένα παράδειγμα για την περίπτωση όπου *n=3* και *m=2*.

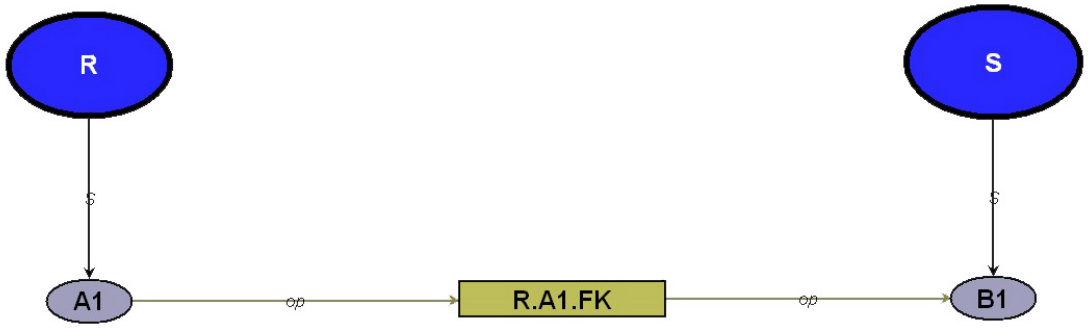


Εικόνα 2.2. Αναπαράσταση περιορισμού πρωτεύοντος κλειδιού

2.2.2.2.2 Περιορισμός Ξένου Κλειδιού

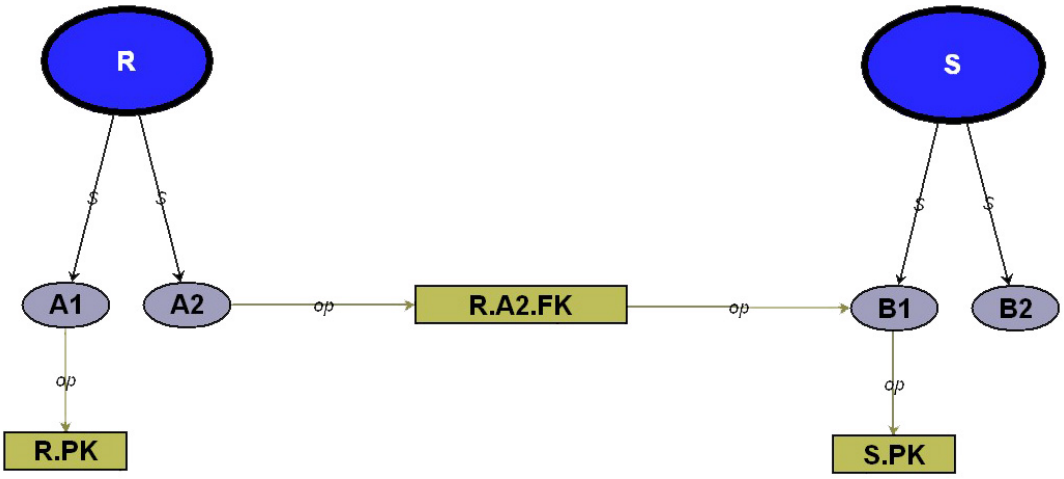
Έστω οι οντότητες R (A_1, A_2, …) και S (B_1, B_2, …). Οι προϋποθέσεις για την ύπαρξη ξένου κλειδιού προσδιορίζουν έναν περιορισμό αναφορικής ακεραιότητας μεταξύ των δύο σχημάτων σχέσεων R και S. Ένα σύνολο γνωρισμάτων FK στο σχήμα σχέσης R είναι ξένο κλειδί της R μία τιμή του FK στην μία πλειάδα είτε εμφανίζεται ως τιμή του πρωτεύοντος κλειδιού της S είτε είναι null. Ο απεικόνιση του περιορισμού ξένου κλειδιού περιλαμβάνει (α) έναν νέο κόμβο τύπου *Περιορισμού*, (β) μία ακμή που ξεκινάει από το γνώρισμα της σχέσης που αναφέρει (έστω το γνώρισμα A_1) και καταλήγει στον κόμβο του περιορισμού και (γ) μία ακμή που ξεκινάει από τον κόμβο περιορισμού και καταλήγει στα κατάλληλα γνωρίσματα που της σχέσης που αναφέρεται. Και οι δύο ακμές χαρακτηρίζονται με *op*. Ο νέος κόμβος τύπου *Περιορισμού* που δημιουργείται έχει ως ετικέτα την τιμή *<Ονομα_οντότητας_που_αναφέρει>.<Όνομα_γνωρίσματος_που_αναφέρει>.FK*.

Στην Εικόνα 2.3 παρουσιάζεται ένα παράδειγμα όπου το γνώρισμα A_1 της οντότητας *R* αναφέρεται στο γνώρισμα B_1 της οντότητας *S*.



Εικόνα 2.3. Αναπαράσταση περιορισμού ξένου κλειδιού

Στην Εικόνα 2.4 παρουσιάζεται ένα πλήρες παράδειγμα όπου απεικονίζονται οι σχέσεις $R(\underline{A_1}, A2)$ και $S(\underline{B_1}, B2)$. Η οντότητα R έχει ως πρωτεύον κλειδί το γνώρισμα A_1 και το γνώρισμα A_2 αναφέρεται στο γνώρισμα $S.B_1$. Επίσης η οντότητα S έχει ως πρωτεύον κλειδί το γνώρισμα B_1.

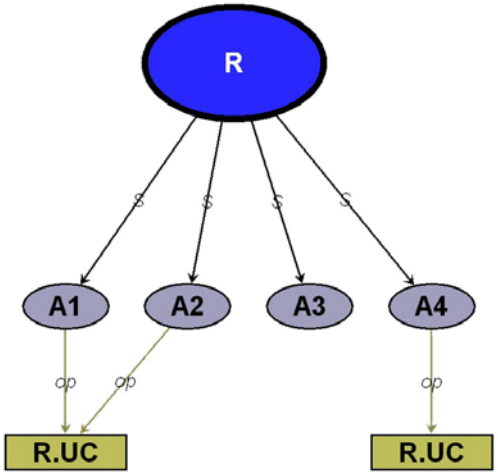


Εικόνα 2.4. Αναπαράσταση περιορισμού πρωτεύοντος και ξένου κλειδιού

2.2.2.2.3 Περιορισμός Μοναδικής Τιμής Γνωρίσματος

Έστω η οντότητα $R(A_1, A_2, \ldots)$ με k περιορισμούς μοναδικής τιμής γνωρίσματος στα γνωρίσματα $A_1, A_2, \ldots, A_m$. Οι περιορισμοί αυτοί αναπαρίστανται ως ένας κατευθυνόμενος γράφος που περιλαμβάνει (α) k νέους κόμβους που αναπαριστούν τους k περιορισμούς μοναδικής τιμής γνωρίσματος και (β) $\sum_{i=1}^{k} u_i$ ακμές όπου u_i το πλήθος των γνωρισμάτων που περιλαμβάνει καθένας από τους k περιορισμούς μοναδικής τιμής γνωρίσματος. Οι ακμές ξεκινούν από τα γνωρίσματα που περιλαμβάνει ο κάθε περιορισμός και καταλήγουν στον αντίστοιχο κόμβο τύπου *Περιορισμός* που δημιουργήθηκε για τον συγκεκριμένο περιορισμό. Οι ακμές είναι τύπου *Περιορισμός* και έχουν ως ετικέτα το γράμμα `op`, ενώ οι κόμβοι που δημιουργούνται για τους περιορισμούς έχουν ως ετικέτα την τιμή *<Όνομα_οντότητας>.UC*.

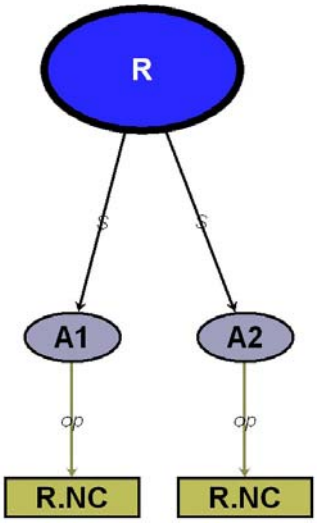
Στην Εικόνα 2.5 παρουσιάζεται ένα παράδειγμα όπου η οντότητα $R(A_1, A_2, A_3, A_4)$ έχει δύο περιορισμούς μοναδικής τιμής γνωρίσματος. Έναν απλό περιορισμό που αφορά το γνώρισμα A_4 και έναν σύνθετο που αφορά τα γνωρίσματα A_1 και A_2.



Εικόνα 2.5. Αναπαράσταση περιορισμού μοναδικής τιμής γνωρίσματος

2.2.2.2.4 Περιορισμός Μη Κενής Τιμής Γνωρίσματος

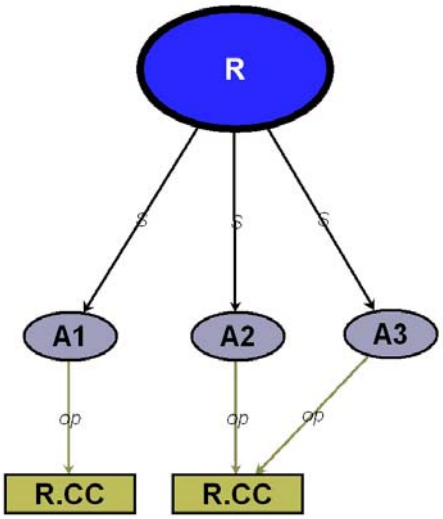
Έστω η οντότητα $R(A_1, A_2, ...)$ με k περιορισμούς μη κενής τιμής γνωρίσματος για τα γνωρίσματα $A_1, A_2, ..., A_m$. Οι περιορισμοί αυτοί αναπαρίστανται ως ένας κατευθυνόμενος γράφος που περιλαμβάνει (α) k νέους κόμβους που αναπαριστούν τους k περιορισμούς μη κενής τιμής γνωρίσματος και (β) k ακμές που κατευθύνονται από τα γνωρίσματα προς τους κόμβους που αναπαριστούν τους περιορισμούς. Οι ακμές έχουν ετικέτα `op` και οι κόμβοι που αναπαριστούν τον περιορισμό έχουν ως ετικέτα την τιμή *<Όνομα_οντότητας>.NC*. Στην Εικόνα 2.6 παρουσιάζεται ένα παράδειγμα όπου η οντότητα $R(A_1, A_2)$ έχει δύο περιορισμούς μη κενής τιμής γνωρίσματος για τα γνωρίσματα A_1 και A_2.



Εικόνα 2.6. Αναπαράσταση περιορισμού μη κενής τιμής γνωρίσματος

2.2.2.2.5 Περιορισμός ελέγχου

Έστω η οντότητα $R(A_1, A_2, ...)$ με k περιορισμούς ελέγχου για τα γνωρίσματα $A_1, A_2, ..., A_m$. Οι περιορισμοί αυτοί αναπαρίστανται ως ένας κατευθυνόμενος γράφος που περιλαμβάνει (α) k νέους κόμβους που αναπαριστούν τους k περιορισμούς ελέγχου και (β) $\sum u_i, i = 1,..,k$ ακμές, όπου u_i είναι το πλήθος των γνωρισμάτων που ελέγχει καθένας από τους k περιορισμούς ελέγχου. Είναι προφανές ότι για σύνθετους περιορισμούς ελέγχου ισχύει ότι $u_i > 1$ ενώ για απλούς περιορισμούς ελέγχου ισχύει $u_i = 1$. Οι ακμές έχουν ετικέτα $\subset$ και οι κόμβοι που αναπαριστούν τον περιορισμό έχουν ως ετικέτα την τιμή *<Όνομα_οντότητας>.CC*. Στην Εικόνα 2.7 παρουσιάζεται ένα παράδειγμα όπου η οντότητα $R(A_1, A_2, A_3)$ έχει δύο περιορισμούς ελέγχου. Ένα απλό περιορισμό ελέγχου για τα γνωρίσματα A_1 και έναν σύνθετο περιορισμό ελέγχου για το γνώρισμα A_3.



Εικόνα 2.7. Αναπαράσταση περιορισμού ελέγχου

2.2.2.3 Ερωτήσεις

Υπάρχει ένας γενικός τρόπος μοντελοποίησης του μεγαλύτερου μέρους των SQL ερωτήσεων. Για λόγους καλύτερης αντίληψης της μοντελοποίησης θα χωρίσουμε τα είδη των ερωτήσεων σε τέσσερις κλάσεις:

- Απλές ερωτήσεις Επιλογής – Προβολής – Συνένωσης (Select-Project-Join, SPJ).
- Ερωτήσεις Επιλογής – Προβολής – Ομαδοποίησης (Select-Project-Group, SPG).
- Εμφωλευμένες ερωτήσεις.
- Ερωτήσεις Συνένωσης πίνακα με τον εαυτό του.

2.2.2.3.1 Ερωτήσεις Επιλογής – Προβολής – Συνένωσης (SPJ)

Η πρώτη κλάση ερωτήσεων περιλαμβάνει ερωτήσεις Επιλογής – Προβολής – Συνένωσης, π.χ. ερωτήσεις με απλές συνθήκες ένωσης οντοτήτων.

Έστω Q_1 μια γενική ερώτηση τύπου SPJ που συμπεριλαμβάνει *n* οντότητες (R_1, R_2, …, R_n). Η σύνταξη της ερώτησης Q_1 είναι η εξής:

```
Q1:   SELECT    R1.A11 as A1, R2.A21 as A2,…, Rn.Anm as Am
      FROM      R1, R2,…, Rn
      WHERE     conj1(R1.A11, constant),…, conjs(Rn.Ank, R1.A1r)
```

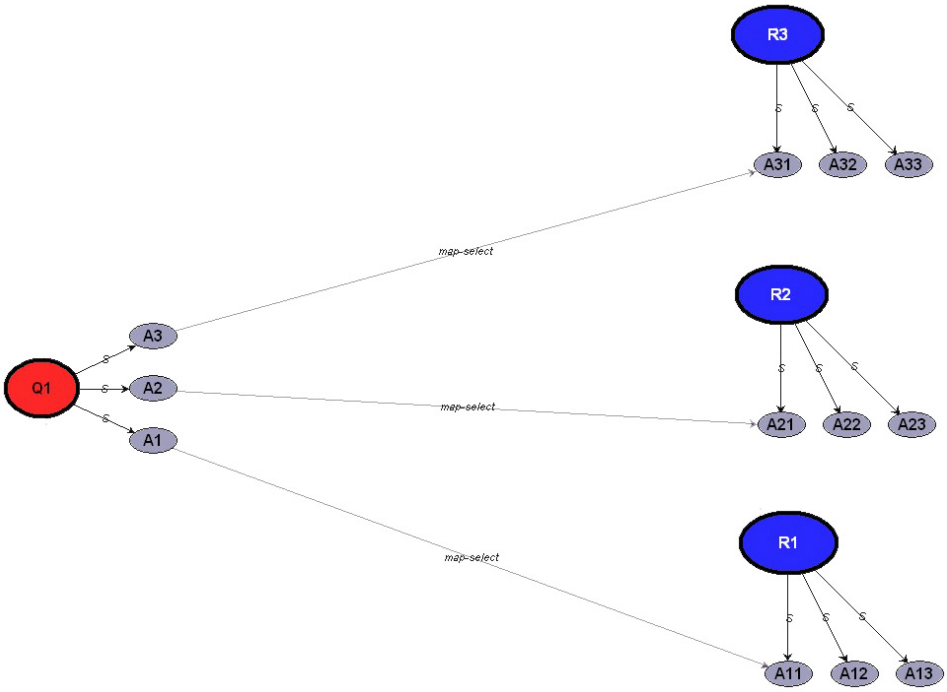
Η μοντελοποίηση με χρήση γράφου της ερώτησης περιλαμβάνει έναν νέο κόμβο που αναπαριστά την ερώτηση Q. Η αναπαράσταση της ερώτησης είναι ένας κατευθυνόμενος γράφος που συνδέει τον κόμβο της ερώτησης *Q* με όλα τα γνωρίσματα σχήματος A_1, A_2,…, A_m. Οι ακμές που συνδέουν τον κόμβο *Q* με όλα τα γνωρίσματα υποδηλώνουν συσχετίσεις σχήματος οπότε έχουν ως ετικέτα το γράμμα S· δηλαδή με τον ίδιο τρόπο με τον οποίο συνδέονται οι κόμβοι οντότητας με τους κόμβους γνωρισμάτων.

Για την αναπαράσταση της σχέσης που υπάρχει ανάμεσα στο δέντρο της ερώτησης και τις υποκείμενες οντότητες θα κάνουμε την παραδοχή ότι κάθε ερώτηση αναλύεται σε τρία κύρια μέρη, το μέρος της *επιλογής* (SELECT), το μέρος *από* (FROM) και το μέρος *όπου* (WHERE). Καθένα από τα μέρη αυτά αναπαρίσταται τελικά σε έναν υπογράφο.

Με βάση τα παραπάνω η αναπαράσταση με χρήση γράφου της παραπάνω ερώτησης SQL είναι η σύνθεση τριών υπογράφων, καθένας από τους οποίους αντιστοιχεί σε ένα από τα τρία μέρη της ερώτησης - SELECT, FROM, WHERE.

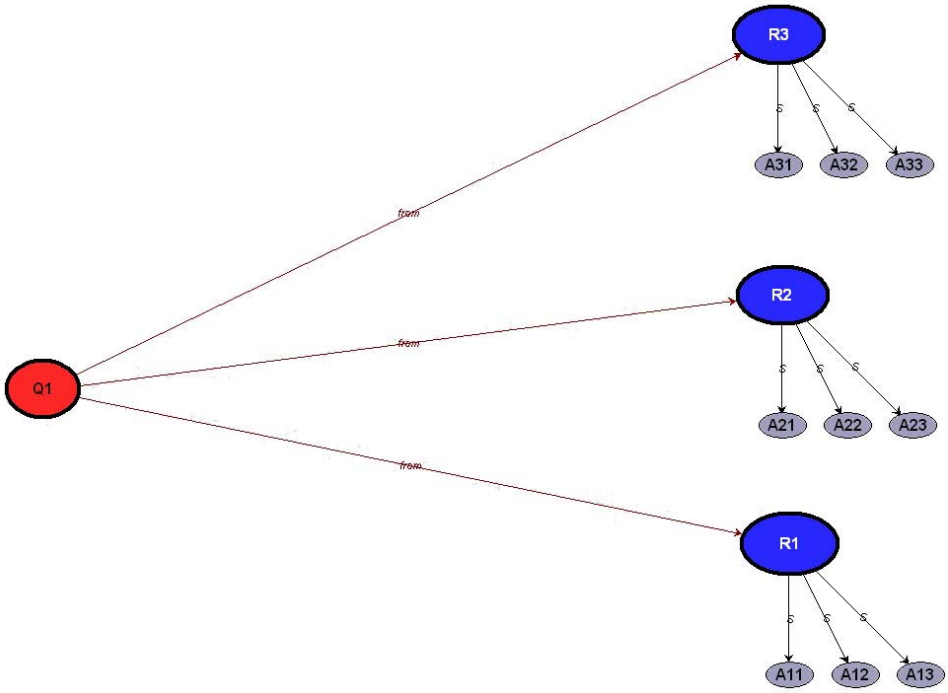
Για την αναπαράσταση των τμημάτων αυτών εισάγεται η ακόλουθη σημειολογία:

- Από την στιγμή που το τμήμα επιλογής (SELECT) μιας ερώτησης περιλαμβάνει μια λίστα γνωρισμάτων, οι τιμές των οποίων θα ανακτηθούν από τα αντίστοιχα γνωρίσματα των οντοτήτων, χρησιμοποιούνται ακμές με ετικέτα <map-select> που κατευθύνονται από τα γνωρίσματα της ερώτησης στα γνωρίσματα της οντότητας. Στην Εικόνα 2.8 παρουσιάζεται η αναπαράσταση του τμήματος SELECT μίας ερώτησης που επιλέγει τρία πεδία από τρεις διαφορετικούς πίνακες.



Εικόνα 2.8. Αναπαράσταση του τμήματος *επιλογή* μιας ερώτησης SPJ

- Το τμήμα *από* (FROM) της ερώτησης μπορεί να θεωρηθεί ως η συσχέτιση της ερώτησης με τις οντότητες που απαιτούνται για την επεξεργασία της ερώτησης. Επομένως για κάθε οντότητα που αναφέρεται στο τμήμα *από*, υπάρχει μια ακμή με ετικέτα `<from>` που συνδέει τον κόμβο της ερώτησης Q με την αντίστοιχη οντότητα. Στην Εικόνα 2.9 παρουσιάζονται τμήμα FROM της ερώτησης Q_1.

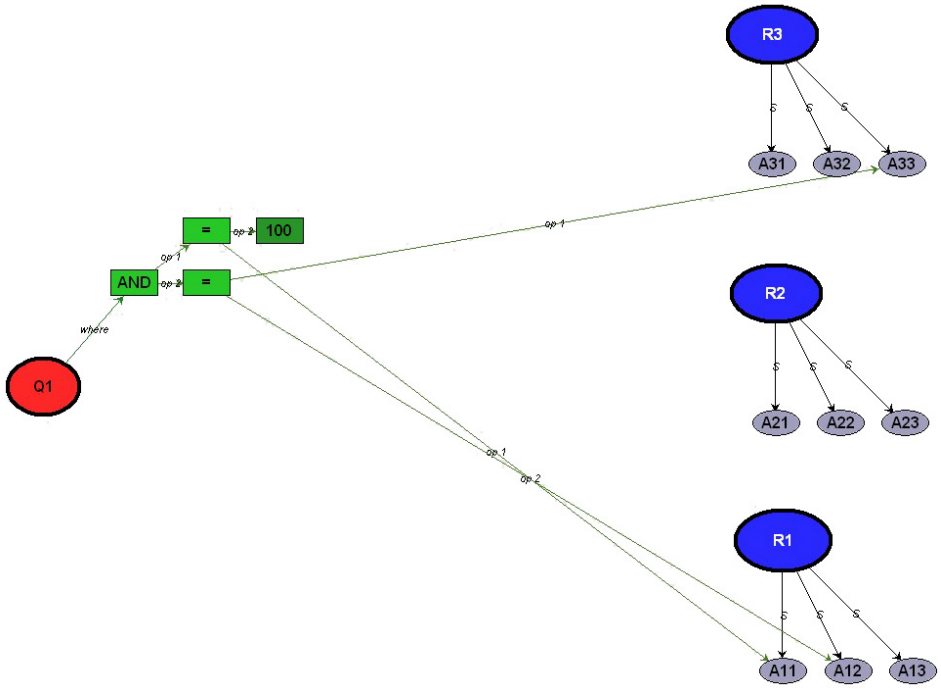


Εικόνα 2.9. Αναπαράσταση του τμήματος *από* μιας ερώτησης SPJ

- Το τμήμα *όπου* (WHERE) της ερώτησης μπορεί να θεωρηθεί ως ο συνδυασμός λειτουργιών (f_1, f_2,…,f_s) καθεμία από τις οποίες μπορεί να περιλαμβάνει ένα η περισσότερα γνωρίσματα των υποκείμενων οντοτήτων. Διακρίνουμε αυτές τις λειτουργίες σε τρεις κύριες κατηγορίες
  - A `op` constant
  - A `op` A'
  - A `IN` Q

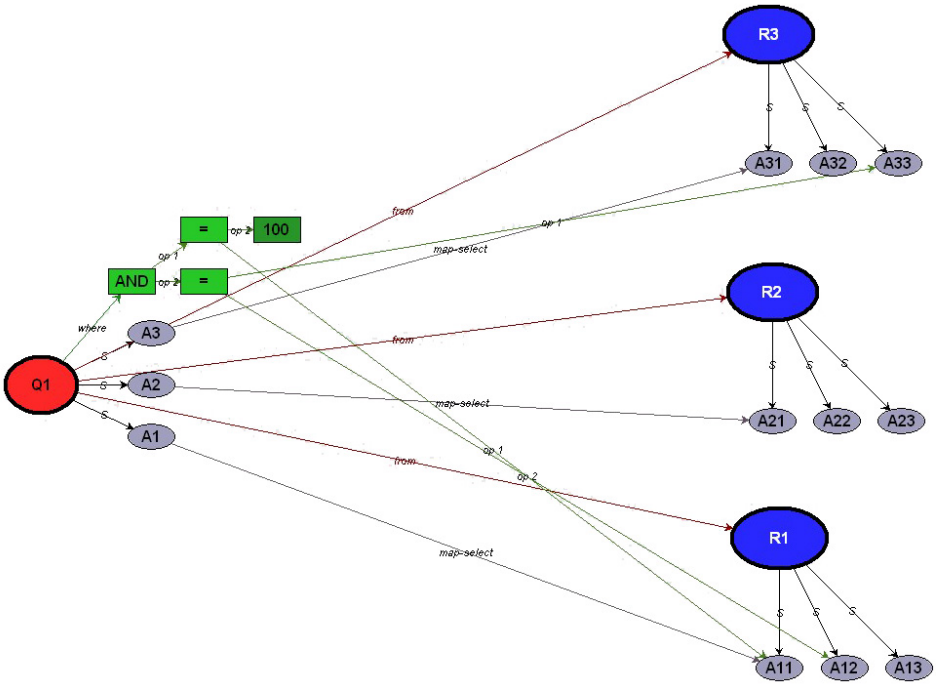
  όπου A, A' είναι γνωρίσματα των υποκείμενων οντοτήτων, `op` είναι ένας δυαδικός τελεστής (π.χ. <, >, =, ≤, ≥, != κτλ).

  Η αναπαράσταση των λειτουργιών $\texttt{conj}_i$ του τμήματος *όπου* μιας ερώτησης μπορεί να θεωρηθεί ως μια τριαδική σχέση ανάμεσα στην ερώτηση και τους δύο τελεστές που εμπλέκονται στην λειτουργία. Η ερώτηση ορίζει την λειτουργία που θα πραγματοποιηθεί ανάμεσα στους δύο τελεστές μέσω του κόμβου τύπου *Δυαδικού Τελεστή*. Οι ακμές με ετικέτα `<where>` έχουν τέτοιο προσανατολισμό ώστε να η αναπαράσταση να είναι ορθή. Στην Εικόνα 2.10 παρουσιάζεται το τμήμα WHERE της ερώτησης Q_1.



Εικόνα 2.10. Αναπαράσταση του τμήματος *όπου* μιας ερώτησης SPJ

Ολόκληρος ο γράφος που προκύπτει από την μοντελοποίηση μιας απλής ερώτησης Επιλογής – Προβολής – Ένωσης, προκύπτει από την σύνθεση των αναπαραστάσεων των τμημάτων *Επιλογή*, *Από*, *Όπου* και παρουσιάζεται στην Εικόνα 2.11.



Εικόνα 2.11. Αναπαράσταση SPJ ερώτησης SQL

2.2.2.3.2 Ερωτήσεις Επιλογής – Προβολής – Ομαδοποίησης (SPG)

Η δεύτερη κλάση ερωτήσεων SQL περιλαμβάνει τις ερωτήσεις Επιλογής – Προβολής – Ομαδοποίησης (Select – Project – Group, SPG), π.χ. ερωτήσεις που περιέχουν συναθροιστικές συναρτήσεις.

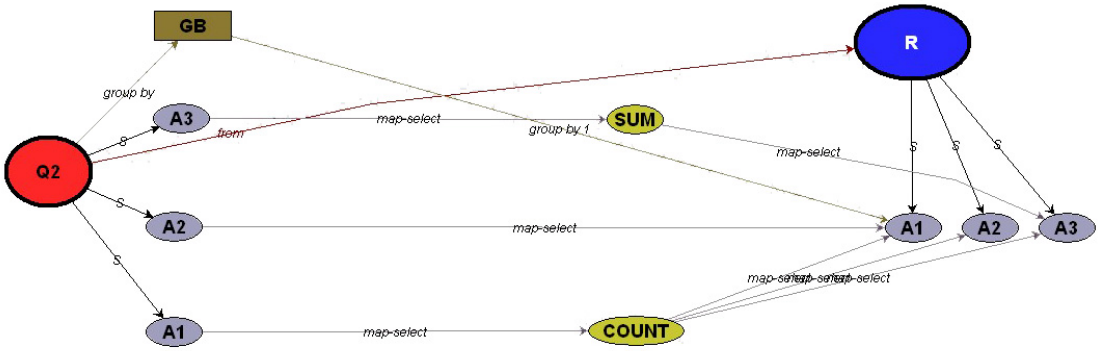
Έστω Q_2 μια γενική ερώτηση τύπου SPG που συμπεριλαμβάνει *n* οντότητες (R_1, R_2, …, R_n). Η σύνταξη της ερώτησης Q_2 είναι η εξής:

```
Q2:   SELECT     COUNT (R.*) AS A1, R.A1 AS A2, SUM(R.Am) AS Am
      FROM       R
      GROUP BY   R.A1
```

Για την αναπαράσταση των SPG ερωτήσεων εισάγονται δύο νέοι τύπους κόμβων: (α) ένας κόμβος τύπου *Ομαδοποίηση Κατά* με ετικέτα GB που θα συνδεθεί με τα γνωρίσματα ομαδοποίησης και (β) ένας κόμβος τύπου *Συνάρτησης* για κάθε συναθροιστική συνάρτηση (π.χ. SUM, COUNT, MIN) με ετικέτα το όνομα της αντίστοιχης συνάρτησης. Για τα γνωρίσματα εκείνα με βάση τα οποία ομαδοποιούνται τα δεδομένα, εισάγουμε μία ακμή τύπου *Ομαδοποίηση Κατά* που κατευθύνεται από τον κόμβο GB και καταλήγει στα αντίστοιχα γνωρίσματα. Η ετικέτα των συγκεκριμένων ακμών είναι `<GROUP BY-n>` όπου *n* η σειρά του κάθε γνωρίσματος στην λίστα των γνωρισμάτων ομαδοποίησης. Επίσης έχουμε εισάγει μια ακμή τύπου *Ομαδοποίηση Κατά* με ετικέτα `<GROUP_BY>` η οποία ξεκινάει από τον κόμβο της ερώτησης *Q* και καταλήγει στον κόμβο GB. Κάθε κόμβος τύπου *Συνάρτησης*

συνδέεται με μια ακμή τύπου *Αντιστοίχισης* με το γνώρισμα εκείνο της ερώτησης όπου προβάλλονται τα αποτελέσματα της. Τα γνωρίσματα που χρησιμοποιούνται ως ορίσματα μίας συναθροιστικής συνάρτησης ενώνονται με την αντίστοιχη συνάρτηση με μια ακμή τύπου *Αντιστοίχισης* αφού η σχέση τους υποδεικνύει την αντιστοίχιση ενός γνωρίσματος της ερώτησης στο αντίστοιχο γνώρισμα μιας οντότητας μέσω της συναθροιστικής συνάρτησης.

Στην Εικόνα 2.12 παρουσιάζεται η γραφική αναπαράσταση της ερώτησης Επιλογής – Προβολής – Ομαδοποίησης Q_2 για $m=3$.



Εικόνα 2.12. Αναπαράσταση ερώτησης SPG

2.2.2.3.3 Εμφωλευμένες ερωτήσεις

Η τρίτη κλάση ερωτήσεων περιλαμβάνει τις εμφωλευμένες συναρτήσεις, δηλαδή τις ερωτήσεις που έχουν πλήρεις προτάσεις *SELECT…FROM…WHERE* στο τμήμα *WHERE* μιας άλλης ερώτησης,

Έστω η εμφωλευμένη ερώτηση Q_3:

```
Q3:   SELECT R1.A1 AS A1
      FROM R1
      WHERE R1.A2 IN
            Q':   (SELECT R2.B1 AS B1
                   FROM R2
                  )
```

Για την αναπαράσταση των εμφωλευμένων ερωτήσεων θεωρείται ότι η μοντελοποίηση εμφωλευμένων ερωτήσεων είναι μια συγκεκριμένη περίπτωση ερωτήσεων *Επιλογής – Προβολής – Ένωσης* (SPJ) όπου το τμήμα *where* έχει ειδική μορφή όπως περιγράφηκε προηγουμένως.

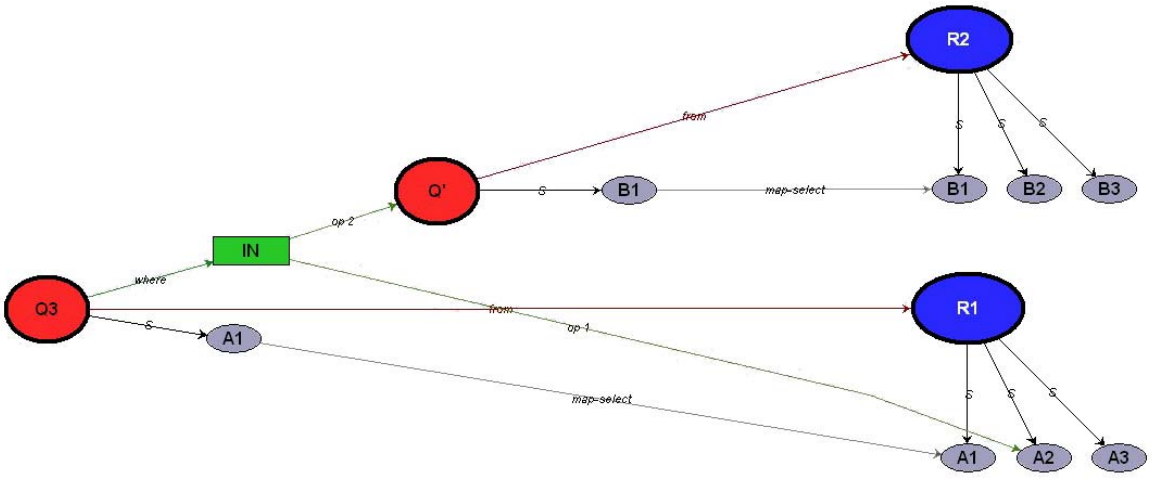
Στην ειδική περίπτωση που έχουμε εμφωλευμένη ερώτηση η λειτουργία που περιλαμβάνεται στο τμήμα *where* της ερώτησης ανήκει στο ακόλουθο είδος:

- A op Q

όπου A είναι ένα γνώρισμα των εμπλεκομένων οντοτήτων, Q είναι η εμφωλευμένη ερώτηση και op είναι ένας δυαδικός τελεστής όπως οι IN, EXISTS, ANY, κτλ.

Η σχέση που υποδηλώνει ο τελεστής op ανάμεσα στα γνωρίσματα της εξωτερικής ερώτησης και τα γνωρίσματα της εμφωλευμένης ερώτησης, αναπαριστάται με έναν νέο κόμβο, που έχει ως ετικέτα το όνομα του δυαδικού τελεστή (π.χ. IN, EXISTS, ANY). Ο νέος κόμβος είναι τύπου *Τελεστής*, δηλαδή είναι ο ίδιος κόμβος που χρησιμοποιείται για την αναπαράσταση του τμήματος *where* μίας ερώτησης *Επιλογής – Προβολής – Ένωσης*.

Στην Εικόνα 2.13 παρουσιάζεται η γραφική αναπαράσταση μιας εμφωλευμένης ερώτησης :



Εικόνα 2.13. Αναπαράσταση εμφωλευμένης ερώτησης

2.2.2.3.4 Ερωτήσεις Συνένωσης πίνακα με τον εαυτό του

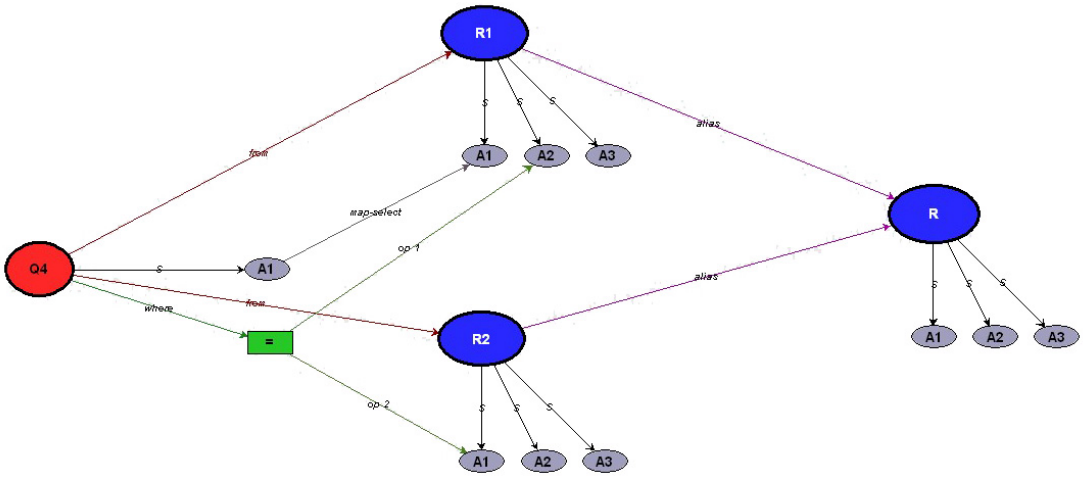
Η τέταρτη κλάση ερωτήσεων περιλαμβάνει τις ερωτήσεις συνένωσης πίνακα με τον εαυτό του, δηλαδή ερωτήσεις που εκτελούν μια λειτουργία συνένωσης οντοτήτων που περιλαμβάνει δύο ή περισσότερες φορές την ίδια οντότητα.

Έστω Q_4 επερώτηση όπου ο πίνακας R_1 συνενώνεται με τον εαυτό του:

```
Q4:   SELECT R1.A1 AS A1
      FROM R AS R1, R AS R2
      WHERE R1.A2 = R2.A1
```

Για την απεικόνιση αυτής της κλάσης ερωτήσεων εισάγεται ένας νέος τύπος ακμών, οι ακμές τύπου *ψευδωνύμου* με ετικέτα `<alias>`. Οι συγκεκριμένες ακμές χρησιμοποιούνται για τη σύνδεση της πηγαίας οντότητας με τις μετονομασμένες οντότητες όπως αναφέρονται στο τμήμα `FROM` της ερώτησης. Η ακμές *ψευδωνύμου* ξεκινούν από τον κόμβο των μετονομασμένων οντοτήτων και καταλήγουν στον κόμβο της πηγαίας οντότητας. Για την απεικόνιση των μετονομασμένων οντοτήτων αντιγράφουμε την πηγαία οντότητα με όλα τα γνωρίσματα της και τα τοποθετούμε ανάμεσα στις ερωτήσεις και τις οντότητες. Στη συνέχεια απεικονίζονται τα τμήματα `SELECT`, `FROM` και `WHERE` της ερώτησης που συνδέονται όπως είναι αναμενόμενο με τις μετονομασμένες οντότητες και όχι με την πηγαία.

Στην Εικόνα 2.14 παρουσιάζεται η αναπαράσταση της ερώτησης Q_4:



Εικόνα 2.14. Αναπαράσταση ερώτησης συνένωσης πίνακα με τον εαυτό του

2.2.2.4 Όψεις

Η αναπαράσταση μίας όψης μπορεί να θεωρηθεί ότι ταυτίζεται με την αναπαράσταση μιας ερώτησης SQL αφού όπως παρουσιάστηκε στον ορισμό μιας όψης, αυτή προκύπτει από μια επερώτηση.

2.3 Αναπαράσταση της εξέλιξης σχήματος βάσης δεδομένων.

Στην ενότητα αυτή, παρουσιάζεται ένα σύνολο κανόνων που επιτρέπουν την εύρεση της επίδρασης που έχει μια αλλαγή στο σχήμα της βάσης δεδομένων πάνω στα υπόλοιπα στοιχεία του σχήματος της βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν. Επίσης παρουσιάζεται ένας αυτοματοποιημένος τρόπος απόκρισης σε αυτές τις αλλαγές. Το αντίκτυπο των αλλαγών επηρεάζει το σύστημα με δυο τρόπους:

- Συντακτικά: μια αλλαγή μπορεί να εγείρει ένα σφάλμα μεταγλώττισης ή εκτέλεσης κατά την εκτέλεση ενός τμήματος κώδικα
- Σημασιολογικά: μια αλλαγή μπορεί να έχει επίδραση στη σημασιολογία του συστήματος.

Το σύνολο των κανόνων που θα παρουσιαστούν εμπλουτίζουν το γράφο με τις ενέργειες που θα πρέπει να γίνουν όταν λάβει χώρα ένα γεγονός αλλαγής στο σχήμα της βάσης. Ο συνδυασμός των γεγονότων και των επισημάνσεων καθορίζει την πολιτική που θα πρέπει να ακολουθηθεί για το χειρισμό μιας πιθανής αλλαγής. Βάση της πολιτικής κάθε κόμβου και των αλληλεξαρτήσεων που υπάρχουν μεταξύ τους βρίσκεται ο αντίκτυπος της αλλαγής στο σύστημα [PVSV07].

2.3.1 Το γενικό πλαίσιο εργασίας για την εξέλιξη του σχήματος της βάσης δεδομένων.

Ο κύριος μηχανισμός για το χειρισμό της εξέλιξης του σχήματος της βάσης δεδομένων είναι η επισήμανση των κόμβων του γράφου με στοιχεία που διευκολύνουν την ανάλυση της επίδρασης των πιθανών γεγονότων πάνω στο σύστημα. Κάθε στοιχείο του σχήματος της βάσης (κόμβος) εμπλουτίζεται με πολιτικές που επιτρέπουν στο σχεδιαστή να καθορίσει τη συμπεριφορά του επισημασμένου κόμβου οποτεδήποτε συμβούν γεγονότα που μετασχηματίζουν το σχήμα της βάσης δεδομένων. Ο συνδυασμός ενός γεγονότος με μια πολιτική χειρισμού αυτού που έχει καθοριστεί από τον σχεδιαστή/διαχειριστή της βάσης προκαλεί την εκτέλεση της κατάλληλης ενέργειας, η οποία, είτε εμποδίζει το γεγονός, είτε μετασχηματίζει το γράφο, ούτως ώστε να ενσωματώσει την προτεινόμενη αλλαγή.

Τα πιθανά γεγονότα που μπορούν να λάβουν χώρα στο σχήμα μιας βάσης δεδομένων είναι δέκα τύπων:

- Προσθήκη ενός γνωρίσματος
- Διαγραφή ενός γνωρίσματος
- Μετονομασία ενός γνωρίσματος
- Αλλαγή του πεδίου ορισμού ενός γνωρίσματος
- Προσθήκη ενός περιορισμού
- Διαγραφή ενός περιορισμού
- Αλλαγή ενός περιορισμού
- Προσθήκη μιας σχέσης
- Διαγραφή μιας σχέσης
- Μετονομασία μιας σχέσης

Ο Πίνακας 2.3 παρουσιάζει τις επιτρεπόμενες επισημάνσεις των στοιχείων του γράφου για κάθε είδος γεγονότος. Στον πίνακα με R παριστάνονται οι σχέσεις, με Ω τα γνωρίσματα, με Q τα ερωτήματα, με C οι περιορισμοί και με V οι όψεις.

		Κόμβοι στους οποίους ορίζεται το γεγονός					**Κόμβοι στους οποίους ορίζονται πολιτικές**				
Γεγονός στο σχήμα της βάσης δεδομένων		**R**	**Ω**	**V**	**C**	Q	**R**	**Ω**	**V**	**C**	Q
Προσθήκη	Ω	√					√		√		√
	C		√								
	R										
Διαγραφή	Ω		√				√	√	√		√
	C				√						
	R	√					√		√		√
Αλλαγή/ Μετονομασία	Ω		√				√	√	√	√	√
	C				√						
	R	√					√		√		√
Αλλαγή πεδίου ορισμού	Ω		√				√	√	√		√

Πίνακας 2.3. Τα πιθανά γεγονότα που μπορεί να συμβούν στο σχήμα της βάσης, οι κόμβοι στους οποίους ορίζεται κάθε γεγονός και πολιτικές χειρισμού του γεγονότος αυτού.

Για κάθε ένα από τα παραπάνω γεγονότα, ο διαχειριστής της βάσης χαρακτηρίζει τους κόμβους του γράφου που επηρεάζονται από το γεγονός με πολιτικές που υπαγορεύουν τον τρόπο με τον οποίο θα διευθετηθεί η αλλαγή. Ορίζονται δύο είδη πολιτικών:

- Προώθηση – Η αλλαγή που έγινε προωθείται, δηλαδή γίνονται οι απαραίτητες αλλαγές στο γράφο, ώστε να λαμβάνει υπόψη του την αλλαγή που έγινε.
- Παρεμπόδιση – Η αλλαγή που έγινε εμποδίζεται, δηλαδή γίνονται οι απαραίτητες αλλαγές στο γράφο, ώστε να διατηρεί την παλιά σημασιολογία του.

Ο ορισμός πολιτικών σε κάθε κόμβο του συστήματος που επηρεάζεται από το συγκεκριμένο γεγονός περιλαμβάνει και την επισήμανσή του στο γράφο στο πλαίσιο εργασίας.

Το παρόν πλαίσιο εργασίας καθορίζει τη συμπεριφορά των στοιχείων του συστήματος που επηρεάζονται από την υποθετική αλλαγή στο σχήμα της βάσης, βάσει της επισήμανσής τους με πολιτικές. Υποθέτοντας ότι η πολιτική κάθε κόμβου είναι να προωθήσει την αλλαγή, η αντιστοιχία μεταξύ των υπό εξέταση αλλαγών στο σχήμα της βάσης και του πως οι κόμβοι του συστήματος (αυτοί που σχετίζονται άμεσα με τον κόμβο στον οποίο ορίστηκε το γεγονός) επηρεάζονται από την αλλαγή παρουσιάζονται στον Πίνακα 2.4. Στην περίπτωση που η πολιτική ενός κόμβου είναι να εμποδίσει την αλλαγή, τότε απλώς αυτός ο κόμβος και ότι εξαρτάται από αυτόν δεν επηρεάζεται με κανένα τρόπο. Στον πίνακα με τη μορφή X.Y παριστάνεται το Y στοιχείο του X στοιχείου.

Ο μηχανισμός που καθορίζει τη συμπεριφορά του γράφου σε μια αλλαγή περιγράφεται με τον Αλγόριθμο Προώθησης Αλλαγών (Propagate Changes - PS), που παρουσιάζεται στον Πίνακα 2.5.

Δοθέντων ενός γράφου και ενός γεγονότος, ο αλγόριθμος ξεκινάει από τον κόμβο που αφορά το γεγονός, θέτει την κατάστασή με βάση την τρέχουσα πολιτική και βρίσκει τις εισερχόμενες ακμές, τις οποίες βάζει σε μια ουρά. Η τρέχουσα πολιτική βρίσκεται με τον αλγόριθμο καθορισμού πολιτικής, βάσει της πολιτικής του τρέχοντος κόμβου και της προηγούμενης πολιτικής (η προκαθορισμένη πολιτική όταν ξεκινάει ο αλγόριθμος είναι προώθηση της αλλαγής).

		Κόμβοι του συστήματος που επηρεάζονται από το γεγονός					
Γεγονός στο σχήμα της βάσης		**R/V**	**R/V.Ω**	**R/V.C**	**Q/V**	**Q/V.Ω**	**Q/V.C**
Add	Ω	Προς προσθήκη κόμβου-παιδιού			Προς προσθήκη κόμβου-παιδιού		
	C		Προς προσθήκη κόμβου-παιδιού				
	R						
Delete	Ω	Επηρεα-σμένος	Προς διαγραφή	Προς διαγραφή	Επηρεα-σμένος	Προς διαγραφή	Προς διαγραφή
	C	Επηρεα-σμένος	Επηρεα-σμένος		Επηρεα-σμένος	Επηρεα-σμένος	
	R	Προς διαγραφή	Προς διαγραφή	Προς διαγραφή	Επηρεα-σμένος	Προς διαγραφή	Προς διαγραφή
Modify/ Rename	Ω	Επηρεα-σμένος	Επηρεα-σμένος		Επηρεα-σμένος	Επηρεα-σμένος	
	C	Επηρεα-σμένος	Επηρεα-σμένος	Επηρεα-σμένος	Επηρεα-σμένος	Επηρεα-σμένος	Επηρεα-σμένος
	R	Επηρεα-σμένος			Επηρεα-σμένος		
Modify domain	Ω	Επηρεα-σμένος	Επηρεα-σμένος		Επηρεα-σμένος	Επηρεα-σμένος	

Πίνακας 2.4. Τα πιθανά γεγονότα που μπορεί να συμβούν στο σχήμα της βάσης και το πώς οι κόμβοι επηρεάζονται από αυτά, με την υπόθεση ότι η πολιτική κάθε κόμβου είναι να προωθήσει την αλλαγή.

Αλγόριθμος Προώθησης Αλλαγών ***(PS)***

Είσοδος: ένας γράφος Go=(Vo,Eo) που αναπαριστά μια βάση δεδομένων και τα ερωτήματα προς αυτή και ένα γεγονός τα event, τύπου eventType που αφορά τον κόμβο eventNode.
Έξοδος: η κατάσταση κάθε κόμβου
Μεταβλητές: μια ουρά queue και ένα μήνυμα msg με:
- το γεγονός event
- τον κόμβο που έστειλε το μήνυμα ns
- τον κόμβο που έλαβε το μήνυμα nr
- τον τύπο της ακμής που ενώνει τον κόμβο αποστολέα και τον κόμβο παραλήπτη edgeType
- την πολιτική του κόμβου αποστολέα p0

Αρχή
queue = νέα ουρά
msg = νέο γεγονός
msg.ns = null
msg.nr = enentNode
msg.event = event
msg.edgeType = null
msg.p0=προώθηση
προώθησηΑλλαγών(msg,queue)
Τέλος

προώθησηΑλλαγών(msg,queue)
Αρχή
p=καθορισμόςΠολιτικής(msg.nr.policy,msg.p0)
άλλαξε την κατάσταση του nr βάση
της πολιτικής p και της κατάστασης ns
για κάθε εισερχόμενη ακμή edge στον nr:
{
nmsg = νέο γεγονός
nmsg.ns = nr
nmsg.nr = κόμβος πηγή της edge
nmsg.event = event
nmsg.p0=p
nmsg.edgeType = τύπος της edge
βάλε το μήνυμα στην queue
}
όσο η queue δεν είναι άδεια επανέλαβε:
{
βγάλε το πρώτο μήνυμα fmsg από την ουρά
προώθησηΑλλαγών(fmsg,queue)
}
Τέλος

καθορισμόςΠολιτικής(msg.nr.policy,msg.p0)
Αρχή
Εάν msg.nr.policy== null
επέστρεψε msg.p0
Αλλιώς εάν msg.nr.policy== msg.p0
επέστρεψε msg.p0
Αλλιώς
επικρατούσαΠολιτική(msg.nr.policy,msg.p0)
Τέλος

Πίνακας 2.5. Αλγόριθμος Προώθησης Αλλαγών (PS)

Ο αλγόριθμος αυτός λειτουργεί ως εξής:

- Εάν ο τρέχων κόμβος δεν έχει πολιτική επιστρέφεται η προηγούμενη πολιτική
- Εάν η πολιτική του τρέχοντος κόμβου και η προηγούμενη πολιτική είναι ίδια, επιστρέφεται αυτή η πολιτική.
- Εάν η πολιτική του τρέχοντος κόμβου και η προηγούμενη πολιτική είναι διαφορετικές καλείται η συνάρτηση εύρεσης της επικρατούσας πολιτικής, ο τρόπος λειτουργίας της οποίας περιγράφεται στην επόμενη ενότητα.

Έπειτα, για κάθε στοιχείο της ουράς επαναλαμβάνει τη διαδικασία. Βρίσκει έτσι όλους τους κόμβους που επηρεάζονται από το γεγονός διασχίζοντας το γράφο κατά πλάτος.

Αποδεικνύεται ότι ο αλγόριθμος προώθησης των αλλαγών (PS) τερματίζει.

2.3.2 Επίλυση συγκρούσεων

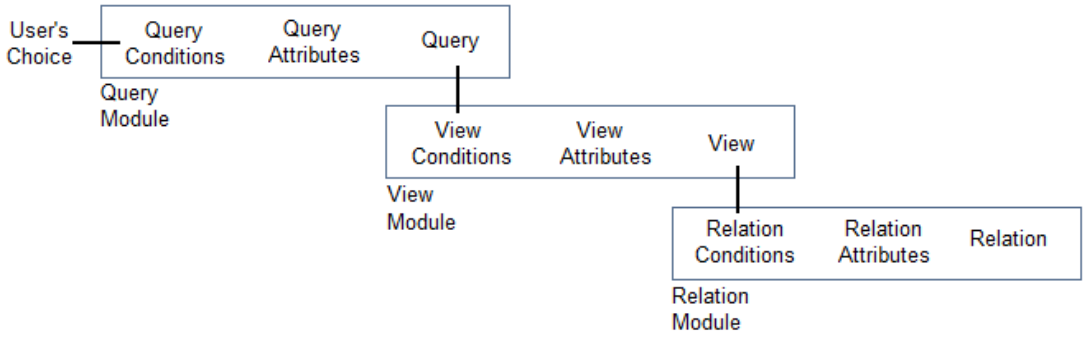
Είναι πιθανό πολιτικές που έχουν οριστεί σε διαφορετικά στοιχεία του γράφου, να μην έχουν πάντα τον ίδιο στόχο. Για παράδειγμα, έστω η περίπτωση που μια όψη V είναι ορισμένη σε μια σχέση R και ένα ερώτημα Q προσπελαύνει την όψη V. Έστω ακόμα ότι η R είναι επισημασμένη με πολιτική τύπου «Προώθηση», π.χ. η τροποποίηση του γράφου επιτρέπεται όταν εισάγεται ένα νέο γνώρισμα στη σχέση:

«Στην περίπτωση εισαγωγής γνωρίσματος τότε προώθησε την αλλαγή»

ενώ η V είναι χαρακτηρισμένη με μια πολιτική τύπου «Παρεμπόδιση» στην περίπτωση που ένα τέτοιο γεγονός, δηλαδή μια εισαγωγή γνωρίσματος, συμβεί σε μια από τις σχέσεις που προσπελαύνει:

«Στην περίπτωση εισαγωγής γνωρίσματος τότε εμπόδισε την αλλαγή»

Ανεξάρτητα από την πολιτική της σχέσης, η εισαγωγή του γνωρίσματος πρέπει να εμποδιστεί στο επίπεδο της όψης και να μην προωθηθεί προς το ερώτημα. Να τονίσουμε ότι εδώ δεν υπάρχει ασυνέπεια: αρχικά, ο σχεδιαστής καθόρισε η σχέση R να προωθεί τις αλλαγές που γίνονται σ' αυτή. Στη συνέχεια, αυτοί που κατά την ανάπτυξη της βάσης κατασκεύασαν την όψη V και το ερώτημα Q, καθόρισαν η όψη V να διατηρεί τη σημασιολογία της, ανεξάρτητα από το τι θα συμβεί στη σχέση R. Την ίδια στιγμή άλλες όψεις που προσπελαύνουν την R μπορεί να επαναπροσαρμόζονται ώστε να αντικατοπτρίζουν τη νέα δομή της R.



Εικόνα 2.15. Ιεραρχία για επίλυση της σύγκρουσης πολιτικών

Η γενική μεθοδολογία για το χειρισμό της σύγκρουσης πολιτικών ακολουθεί τον παρακάτω κανόνα: Όσο υψηλότερα και αριστερότερα είναι ένα στοιχείο του συστήματος, τόσο ισχυρότερη είναι η πολιτική του. Συγκεκριμένα, εκτελούνται τα παρακάτω βήματα:

- Βήμα 1: Όποτε λαμβάνει χώρα μια αλλαγή σε ένα στοιχείο κατώτερου επιπέδου, η αλλαγή εφαρμόζεται στο τμήμα βάσει της πολιτικής του. Καθορίζονται όλα τα στοιχεία υψηλότερου επιπέδου που πιθανώς επηρεάζονται από την αλλαγή. Τα στοιχεία αυτά θα αναφέρονται παρακάτω με τον όρο υποψήφιοι για επαναπροσαρμογή. Η αλλαγή προωθείται κατά την κατεύθυνση των υποψηφίων ή εμποδίζεται.

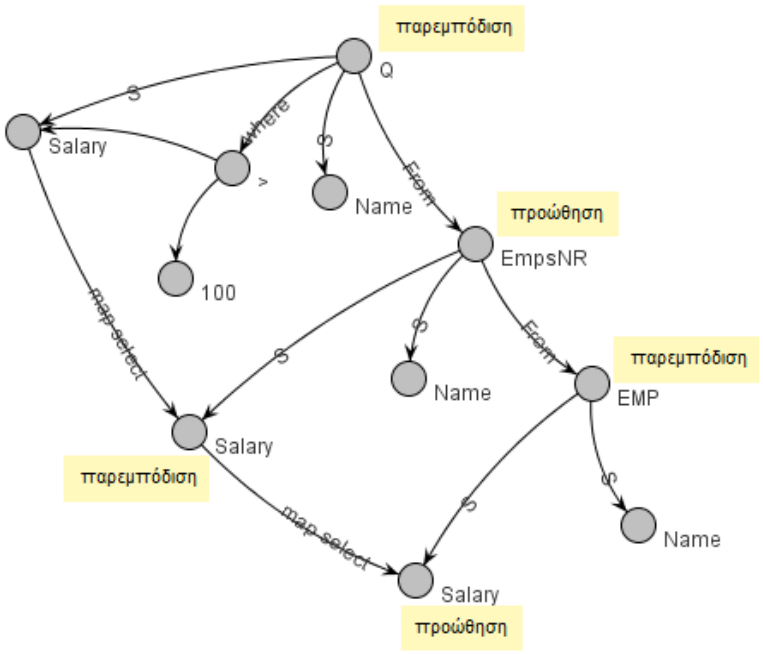
- Βήμα 2: Αν ένα στοιχείο υψηλότερου επιπέδου δεν έχει πολιτική για το χειρισμό της προωθημένης αλλαγής, τότε συμμορφώνεται με την πολιτική που υπαγορεύεται από το κατώτερο επίπεδο. Για παράδειγμα, αν έχει τροποποιηθεί μια όψη και το ερώτημα που την προσπελαύνει δεν έχει πολιτική, τότε το ερώτημα εναναπροσαρμόζεται βάσει της πολιτικής της όψης.
- Βήμα 3: Αν ένα στοιχείο υψηλότερου επιπέδου έχει πολιτική, τότε αυτή υπερσκελίζει την πολιτική που υπαγορεύεται από το στοιχείο του χαμηλότερου επιπέδου. Για παράδειγμα, αν η εισαγωγή ενός γνωρίσματος σε μια σχέση υπαγορεύει την προώθηση της αλλαγής και το ερώτημα που προσπελαύνει τη σχέση έχει πολιτική «Παρεμπόδιση», τότε το ερώτημα δεν αλλάζει, ανεξάρτητα από την πολιτική της σχέσης.
- Βήμα 4: Για τα στοιχεία που ανήκουν στο ίδιο επίπεδο, εξακολουθεί να ισχύει η ίδια μεθοδολογία. Για παράδειγμα, αν η πολιτική για τη διαγραφή γνωρίσματος που ορίζεται σε ένα γνώρισμα υπαγορεύει ότι η αλλαγή πρέπει να εμποδιστεί και η πολιτική για το ίδιο γεγονός που ορίζεται στη σχέση υπαγορεύει ότι η διαγραφή πρέπει να προωθηθεί, τότε η πολιτική του γνωρίσματος υπερσκελίζει την πολιτική της σχέσης.
- Βήμα 5: Επανάληψη της διαδικασίας για το επόμενο στοιχείο.

Έστω, για παράδειγμα, η διάταξη της Εικόνας 2.16. Αυτό το παράδειγμα περιλαμβάνει μια όψη, `EmpsNR`, που χαρακτηρίζει τους υπαλλήλους που είναι κοντά στην ηλικία συνταξιοδότησης: `SELECT * FROM EMP WHERE AGE>60`. Ένα ερώτημα `Q` ορίζεται πάνω σ' αυτή την όψη και επιστρέφει τους υπαλλήλους που είναι κοντά στην ηλικία συνταξιοδότησης που ο μισθός τους είναι υψηλός: `SELECT * FROM EmpsNR WHERE SALARY>100K`. Έστω, επιπλέον, ότι για το γεγονός της διαγραφής του γνωρίσματος `Salary` του `EMP`, έχουν οριστεί πάνω στο γράφο οι ακόλουθες πολιτικές:

- Στο `Salary` της σχέσης `EMP`: Στην περίπτωση διαγραφής του γνωρίσματος `Salary` του `EMP` τότε προώθησε την αλλαγή.
- Στη σχέση `EMP`: Στην περίπτωση διαγραφής του γνωρίσματος `Salary` του `EMP` τότε εμπόδισε την αλλαγή.
- Στο `Salary` της όψης `EmpsNR`: Στην περίπτωση διαγραφής του γνωρίσματος `Salary` του `EMP` τότε εμπόδισε την αλλαγή.
- Στην όψη `EmpsNR`: Στην περίπτωση διαγραφής του γνωρίσματος `Salary` του `EMP` `Salary` του `EMP` τότε προώθησε την αλλαγή.
- Στο ερώτημα `Q`: Στην περίπτωση διαγραφής του γνωρίσματος `Salary` του `EMP` τότε εμπόδισε την αλλαγή.

Οι πολιτική κάθε κόμβου για αυτό το γεγονός φαίνεται στο σχήμα σε κίτρινο πλαίσιο.

Οι παραπάνω πολιτικές ορίζονται σε διαφορετικούς κόμβους του γράφου, χειρίζονται όμως το ίδιο γεγονός. Ακολουθώντας τη μεθοδολογία και λαμβάνοντας υπόψη την ιεραρχία στην Εικόνα 2.15, το αποτέλεσμα της σύγκρουσης των πολιτικών έχει ως εξής:

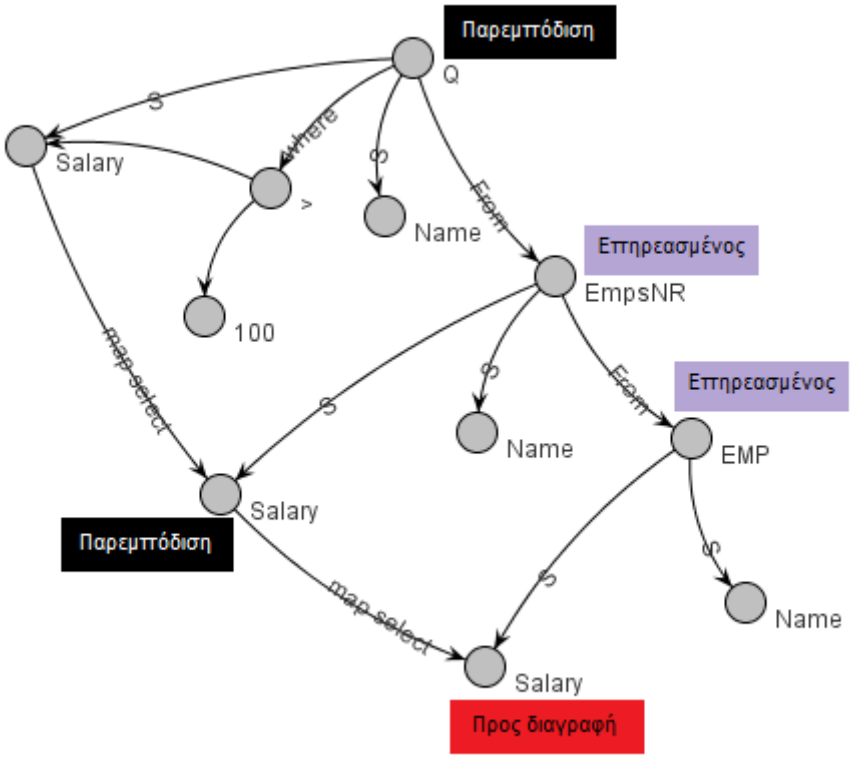


Εικόνα 2.16. Γράφος με ορισμένες πολιτικές.

- Υπάρχει μια σύγκρουση μεταξύ της πολιτικής της σχέσης `EMP`: «Εμπόδισε τις διαγραφές των γνωρισμάτων της σχέσης» και της πολιτικής του γνωρίσματος `Salary`: «Προώθηση». Βάσει της προαναφερθείσας μεθόδου, η πολιτική του γνωρίσματος υπερισχύει έναντι της πολιτικής της σχέσης και προωθεί τη διαγραφή ορίσματος στη σχέση. Η κατάσταση του γνωρίσματος `Salary` της σχέσης `EMP` είναι προς διαγραφή από το γράφο.
- Υπάρχει μια σύγκρουση μεταξύ της πολιτικής του γνωρίσματος `Salary` της σχέσης `EMP`: «Προώθηση» και της πολιτικής του γνωρίσματος `Salary` της όψης `EmpsNR`: «Προώθηση». Βάσει της προαναφερθείσας μεθόδου, η πολιτική του γνωρίσματος της όψης υπερισχύει έναντι της πολιτικής του γνωρίσματος της σχέσης και εμποδίζει τη διαγραφή του γνωρίσματος από την όψη. Ο κόμβος δεν προωθεί την αλλαγή στο γνώρισμα `Salary` του ερωτήματος και στην όψη.
- Η σχέση προωθεί τη διαγραφή στην όψη `EmpsNR`, με την πολιτική της οποίας («Προώθηση») δεν υπάρχει σύγκρουση και η οποία επίσης θέτει την κατάσταση του γνωρίσματος `EmpsNR.Salary` προς διαγραφή από το γράφο.
- Υπάρχει μια σύγκρουση μεταξύ της πολιτικής του γνωρίσματος `Salary` της όψης `EmpsNR`:«Παρεμπόδιση» και της πολιτικής της όψης `EmpsNR`:«Προώθηση». Βάσει της

προαναφερθείσας μεθόδου, η πολιτική του γνωρίσματος της όψης υπερισχύει έναντι της πολιτικής της όψης και εμποδίζει τη διαγραφή του γνωρίσματος από την όψη.

- Το ερώτημα Q ενημερώνεται για τη διαγραφή του ορίσματος από την όψη `EmpsNR` και προσαρμόζει τον εαυτό του σύμφωνα με την πολιτική του: «Παρεμπόδιση». Η πολιτική του γνωρίσματος `Salary` του ερωτήματος δε γίνεται γνωστή αφού δεν έχει ενημερωθεί από το γνώρισμα `Salary` της όψης `EmpsNR` και ο αλγόριθμος τερματίζει εδώ. Οι καταστάσεις των κόμβων έχουν γίνει όπως φαίνεται στην Εικόνα 2.17.



Εικόνα 2.17. Η κατάσταση των κόμβων μετά την επίλυση των συγκρούσεων πολιτικών.

2.3.3 Σκοπός της διπλωματικής

Ο σκοπός της διπλωματικής είναι η δημιουργία ενός λογισμικού συστήματος, του ΕΚΑΤΑΙΟΥ ΙΙ, που θα αναπαριστά μια βάση δεδομένων και τα ερωτήματα προς αυτή με ενιαίο τρόπο ως κατευθυνόμενο γράφο, όπως περιγράφηκε στην παράγραφο 2.2. Το εργαλείο θα δίνει επίσης τη δυνατότητα ορισμού γεγονότων και πολιτικών χειρισμού αυτών των γεγονότων πάνω στους κόμβους του γράφου. Για κάθε γεγονός θα μπορεί βάσει των ορισμένων πολιτικών να βρίσκεται η κατάσταση των κόμβων του γράφου όπως περιγράφηκε στην παράγραφο 2.3. Ο γράφος έχει επιπλέον τη δυνατότητα οπτικοποίησης σε ένα πλάνο, στο οποίο μπορεί να φαίνεται η κατάσταση κάθε κόμβου. Ο διαχειριστής της βάσης

δεδομένων μπορεί να δει έτσι το αντίκτυπο που επιφέρει μια αλλαγή στο σχήμα της βάσης στο όλο σύστημα, να αποφασίσει αν θέλει να την αποδεχτεί ή όχι και αν ναι, να κάνει ευκολότερα και με μικρότερη πιθανότητα λάθους τις κατάλληλες ενέργειες για την ενσωμάτωσή της.

3 Ανάλυση Απαιτήσεων Συστήματος

Σ' αυτό το κεφάλαιο θα περιγραφούν οι απαιτήσεις από την αρχιτεκτονική του λογισμικού συστήματος ΕΚΑΤΑΙΟΣ ΙΙ και στη συνέχεια θα περιγραφούν οι απαιτήσεις για τις λειτουργίες του. Συγκεκριμένα, στην παράγραφο 3.1 παρουσιάζονται τα επιμέρους κύρια τμήματα του συστήματος, μαζί με μια σύντομη περιγραφή των λειτουργιών που το καθένα πρέπει να επιτελεί και στην παράγραφο 3.2 παρουσιάζονται αναλυτικά οι βασικές λειτουργίες που απαιτείται να εκτελεί το κάθε τμήμα και συνεπώς το σύστημα στο σύνολό του.

3.1 Απαιτήσεις από την αρχιτεκτονική του συστήματος

Το σύστημα απαιτείται να έχει τα παρακάτω βασικά τμήματα, καθένα από τα οποία θα έχει ξεχωριστά καθήκοντα:

3.1.1 Ένα τμήμα για αναπαράσταση γράφου

Το τμήμα για την αναπαράσταση πρέπει να υλοποιεί έναν κατευθυνόμενο γράφο, ο οποίος να αναπαριστά με ενιαίο τρόπο το σχήμα μιας βάσης δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν. Πρέπει, δηλαδή, να υλοποιεί τους κόμβους και τις πλευρές του γράφου και το γράφο ως σύνολο αυτών των κόμβων και των πλευρών και των μεταξύ τους σχέσεων.

3.1.2 Ένα τμήμα για εμπλουτισμό του γράφου με σημασιολογία για την εξέλιξη

Απαιτείται η ύπαρξη ενός τμήματος για την υλοποίηση ενός γράφου, ο οποίος θα μπορεί να εμπλουτίζεται με σημασιολογία για την εξέλιξη του σχήματος της βάσης δεδομένων. Αυτό το τμήμα, εκτός από την υλοποίηση του γράφου, πρέπει να προσφέρει επιπλέον δυνατότητες που

αφορούν στην εξέλιξη του σχήματος της βάσης δεδομένων, όπως υλοποίηση των γεγονότων που μπορούν να συμβούν στο σχήμα μιας βάσης δεδομένων (για παράδειγμα, προσθήκη μιας ιδιότητας ή διαγραφή μιας σχέσης), των πολιτικών για τον χειρισμό αυτών των γεγονότων και τρόπου εύρεσης του αντίκτυπου που επιφέρει μια αλλαγή στο σχήμα της βάσης στο σύστημα.

3.1.3 Ένα τμήμα για την οπτικοποίηση του γράφου

Το τμήμα αυτό πρέπει να υλοποιεί έναν γράφο ο οποίος να μπορεί να οπτικοποιηθεί. Όσον αφορά στην εξέλιξη του σχήματος της βάσης δεδομένων, το τμήμα αυτό θα χρησιμοποιεί τις λειτουργίες που προσφέρονται από το τμήμα για τον εμπλουτισμό του γράφου με σημασιολογία για την εξέλιξη.

3.1.4 Ένα τμήμα για τη διεπαφή της εφαρμογής

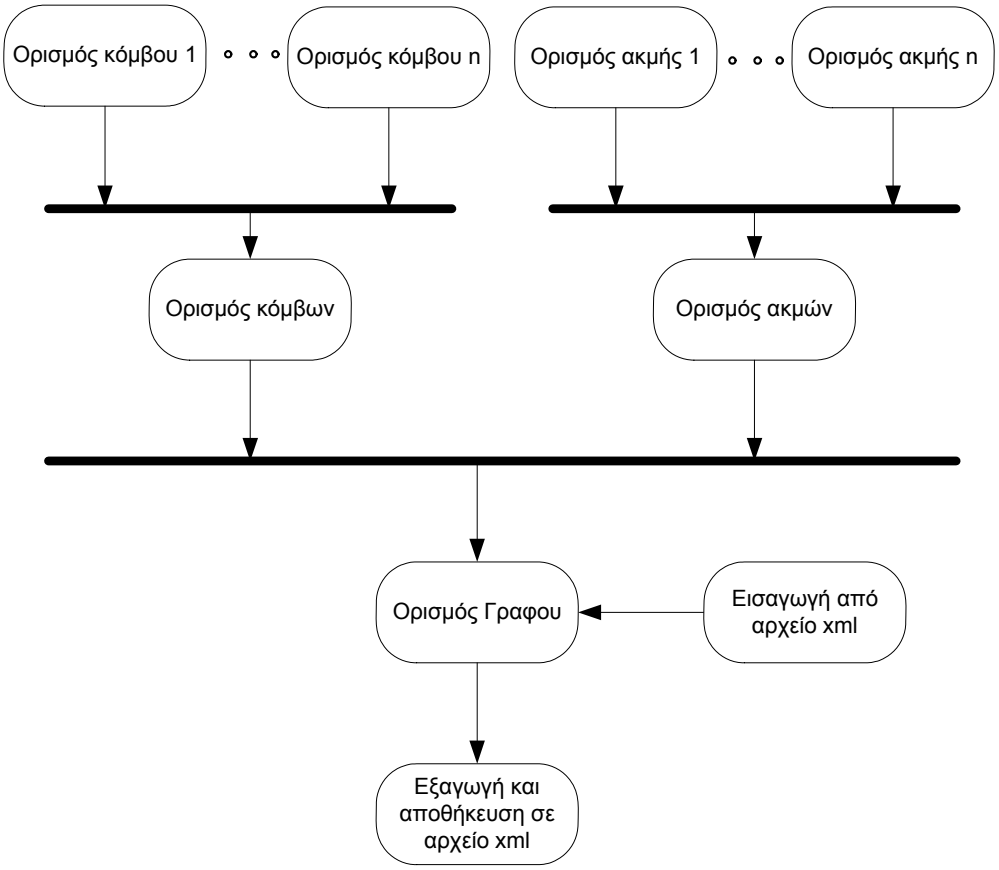
Τέλος, χρειάζεται ένα τμήμα που θα αναλάβει την υλοποίηση της διεπαφής της εφαρμογής με το χρήστη. Το τμήμα αυτό θα εμφανίζει το παράθυρο της εφαρμογής στο οποίο θα οπτικοποιείται ο γράφος και θα παρέχει γραφική πρόσβαση στη λειτουργικότητα του συστήματος.

3.2 Περιγραφή Λειτουργιών

Παρακάτω θα περιγραφούν οι λειτουργίες που απαιτείται να επιτελεί κάθε τμήμα χωριστά.

3.2.1 Λειτουργίες του τμήματος για αναπαράσταση γράφου

Όπως αναφέρθηκε και στην παράγραφο 3.1, το τμήμα για την αναπαράσταση του γράφου πρέπει να υλοποιεί έναν κατευθυνόμενο γράφο, ο οποίος να αναπαριστά με ενιαίο τρόπο το σχήμα μιας βάσης δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν. Αυτό γίνεται με τον ορισμό κόμβων και ακμών (που ορίζονται με βάση τους κόμβους που έχουν ως άκρα) και τον ορισμό του γράφου μέσω αυτών. Πρέπει να δίνεται η δυνατότητα για καθορισμό του ονόματος και του τύπου των κόμβων και των ακμών, καθώς και ο μονοσήμαντος προσδιορισμός τους από έναν μοναδικό αριθμό. Ο γράφος που δημιουργείται δε θα είναι στατικός, αλλά πρέπει να μπορεί να μεταβληθεί προσθέτοντας ή αφαιρώντας κόμβους ή ακμές. Επίσης ένας γράφος θα μπορεί να δημιουργηθεί από το μηδέν προσθέτοντας κόμβους και ακμές ή να εισαχθεί από ένα αρχείο τύπου xml. Ο γράφος θα μπορεί επίσης να εξαχθεί σε ένα αρχείο xml. Οι λειτουργίες του τμήματος απεικονίζονται στο διάγραμμα ροής της Εικόνας 3.1.



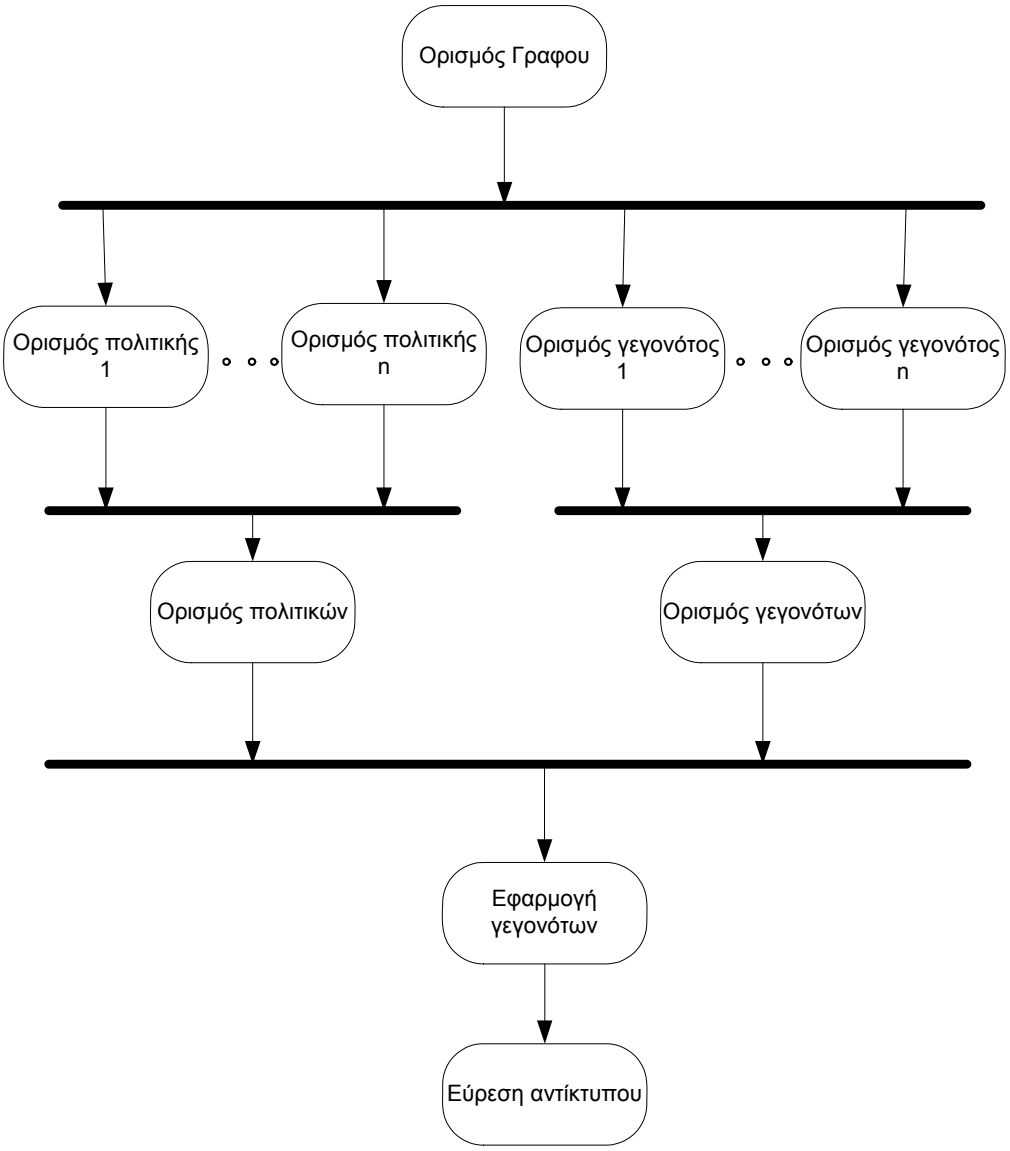
Εικόνα 3.1. Διάγραμμα Ροής Δεδομένων για το τμήμα που είναι υπεύθυνο για την αναπαράσταση του γράφου.

3.2.2 Λειτουργίες του τμήματος για εμπλουτισμό του γράφου με σημασιολογία για την εξέλιξη

Το τμήμα αυτό απαιτείται να προσφέρει, όπως προαναφέρθηκε, επιπλέον δυνατότητες όσον αφορά τον ορισμό γεγονότων και πολιτικών.

Κάθε κόμβος του γράφου θα μπορεί να έχει ορισμένες καμία, μία ή περισσότερες πολιτικές, μία για κάθε γεγονός που αφορά ένα συγκεκριμένο κόμβο.

Αν υπάρχουν συγκρούσεις στις πολιτικές, δηλαδή εάν υπάρχουν περισσότεροι του ενός κόμβοι που έχουν πολιτική για το συγκεκριμένο γεγονός που αφορά τον ίδιο κόμβο, θα επιλύονται ώστε να βρεθεί η επικρατούσα πολιτική (για το συγκεκριμένο γεγονός και τον κόμβο που αφορά). Οι βασικές λειτουργίες του τμήματος απεικονίζονται στο διάγραμμα ροής της Εικόνας 3.2.



Εικόνα 3.2. Διάγραμμα Ροής Δεδομένων για το τμήμα που είναι υπεύθυνο για τον εμπλουτισμό του γράφου με σημασιολογία για την εξέλιξη.

3.2.3 Λειτουργίες του τμήματος για την οπτικοποίηση του γράφου

Το τμήμα αυτό θα υποστηρίζει την οπτική αναπαράσταση του γράφου και όσον αφορά στην εξέλιξη του σχήματος της βάσης θα χρησιμοποιεί τις λειτουργίες που παρέχονται από το τμήμα για τον εμπλουτισμό του γράφου με σημασιολογία για την εξέλιξη. Εδώ, οι κόμβοι εκτός από το όνομα, τον τύπο τους και το μοναδικό «κλειδί» τους, θα έχουν καθορισμένη τη θέση τους στο επίπεδο ως ζεύγος συντεταγμένων (x,y). Επίσης θα υπάρχει τρόπος εύρεσης αν έχουν ορισμένη κάποια πολιτική ή κάποιο γεγονός. Κάθε λειτουργία έχει και αντίστοιχο αντίκτυπο στο πλάνο του γράφου.

3.2.4 Λειτουργίες του τμήματος για τη διεπεφή της εφαρμογής

Το τμήμα αυτό προσφέρει το γραφικό περιβάλλον της εφαρμογής, για την προβολή και επεξεργασία γράφων που αναπαριστούν βάση δεδομένων.

Όσον αφορά τους κόμβους και τις ακμές που συνθέτουν το γράφο, ο χρήστης θα έχει μέσω του γραφικού περιβάλλοντος τις παρακάτω δυνατότητες:

- Θα μπορεί γραφικά να προσθέσει ή/και αφαιρέσει κόμβους/ακμές.
- Θα μπορεί γραφικά να αφαιρέσει ένα σύνολο κόμβων.
- Θα μπορεί να ορίσει όνομα και τύπο για κόμβους και τις ακμές κατά τη δημιουργία τους ή να αλλάξει τον τύπο και το όνομά τους οποιαδήποτε στιγμή.
- Θα μπορεί να καθορίζει τη θέση των κόμβων του γράφου.

Όσον αφορά το γράφο στο σύνολό του, θα δίνονται οι δυνατότητες:

- Εισαγωγής του γράφου από ένα xml αρχείο και οπτικοποίησή του. Σ' αυτήν την περίπτωση (σε αντίθεση με την περίπτωση εισαγωγής υπογράφου από xml αρχείο που αναφέρεται παρακάτω) αν υπάρχει ήδη γράφος στο πλάνο, τότε οποιαδήποτε πληροφορία υπήρχε για αυτόν σβήνεται, όπως και το σχέδιό του από το πλάνο, ενώ οπτικοποιείται ο νέος γράφος και εισάγονται οι πληροφορίες γι' αυτόν.
- Εισαγωγής υπογράφου από ένα xml αρχείο. Σ' αυτήν την περίπτωση αν υπάρχει ήδη γράφος στο πλάνο, τότε αυτός παραμένει ως έχει, ενώ οπτικοποιείται και ο νέος γράφος και εισάγονται οι πληροφορίες γι' αυτόν.
- Εξαγωγής και αποθήκευσης του γράφου που φαίνεται στο πλάνο σε ένα xml αρχείο.
- Εξαγωγής και αποθήκευσης του γράφου που φαίνεται στο πλάνο ως εικόνα σε ένα αρχείο jpg.
- Εκκαθάρισης του πλάνου. Αν υπάρχει γράφος στο πλάνο, τότε οποιαδήποτε πληροφορία υπήρχε για αυτόν σβήνεται, όπως και το σχέδιό του από το πλάνο.

Επιπλέον, όσον αφορά το χειρισμό του γράφου, ο χρήστης θα έχει δύο δυνατότητες:

- Να μετακινήσει το γράφο στο πλάνο.
- Να επιλέξει τα στοιχεία του γράφου.

Θα προσφέρονται επίσης λειτουργίες κλιμάκωσης:

- Μεγέθυνση.
- Σμίκρυνση.

Όσον αφορά τα γεγονότα, ο χρήστης θα μπορεί γραφικά:

- Να ορίσει νέο γεγονός απευθείας σε ένα κόμβο.

- Να προβάλλει όλα τα γεγονότα που έχουν οριστεί για έναν κόμβο.
- Να επιλέξει ένα γεγονός και να το διαγράψει από το σύνολο των γεγονότων αυτού του κόμβου.
- Να ορίσει ένα νέο γεγονός προσδιορίζοντας και τον κόμβο στον οποίο το προσθέτει.
- Να επιλέξει ένα γεγονός και να τονίσει στο πλάνο με κατάλληλο χρώμα το σύνολο των κόμβων και των πλευρών που επηρεάζονται από αυτό το γεγονός. Συγκεκριμένα, όταν ένας κόμβος είναι να διαγραφεί χρωματίζεται κόκκινος, όταν είναι να προστεθεί σ' αυτόν κόμβος-παιδί χρωματίζεται πράσινος και όταν επηρεάζεται με άλλο τρόπο από αυτό το γεγονός χρωματίζεται κίτρινος.

Όσον αφορά τις πολιτικές ο χρήστης θα έχει τη δυνατότητα:

- Να ορίσει νέα πολιτική για ένα γεγονός που αφορά έναν κόμβο απευθείας σε ένα κόμβο.
- Να ορίσει μια πολιτική ταυτόχρονα σε πολλούς κόμβους.
- Να προβάλλει όλες τις πολιτικές που έχουν οριστεί για έναν κόμβο.
- Να επιλέξει μια πολιτική και να τη διαγράψει από το σύνολο των πολιτικών αυτού του κόμβου.

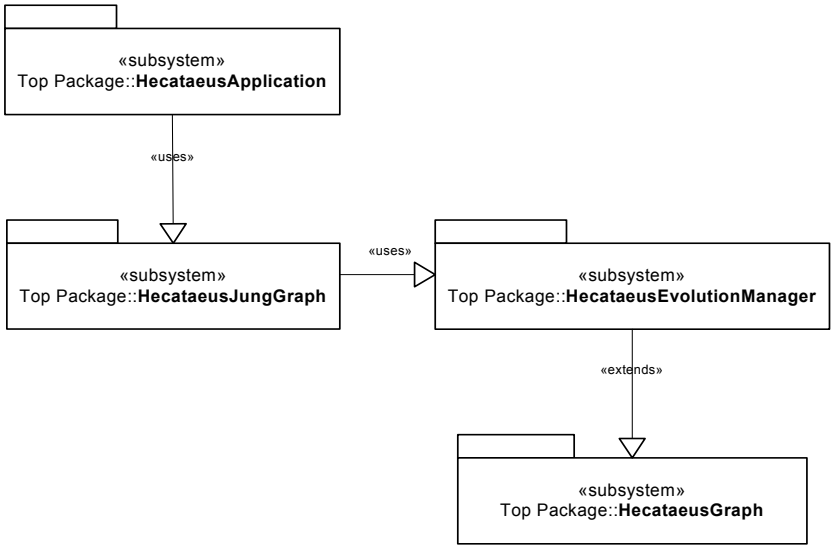
Τέλος, ο χρήστης θα μπορεί να προβάλλει κείμενο-βοήθεια με συνοπτικές οδηγίες για τις λειτουργίες που προσφέρονται.

4 *Σχεδίαση Συστήματος*

Σ' αυτήν την ενότητα παρουσιάζεται η σχεδίαση του συστήματος. Συγκεκριμένα, στην παράγραφο 4.1 παρουσιάζονται τα επιμέρους κομμάτια από τα οποία αποτελείται ο πηγαίος κώδικας του συστήματος. Στην παράγραφο 4.2 περιγράφονται συνοπτικά οι λειτουργίες των κλάσεων.

4.1 Αρχιτεκτονική

Με βάση την ανάλυση των απαιτήσεων όπως έγινε στην ενότητα 3, το σύστημα υλοποιήθηκε από τέσσερα βασικά υποσυστήματα-πακέτα, ένα για κάθε απαιτούμενο τμήμα. Το υποσύστημα HecataeusGraph, το οποίο αντιστοιχεί στο τμήμα για την αναπαράσταση του γράφου, περιλαμβάνει τις κλάσεις για την υλοποίηση ενός γράφου, ως σύνολο κόμβων και ακμών. Το υποσύστημα HecataeusEvolutionManager, το οποίο αντιστοιχεί στο τμήμα για τον εμπλουτισμό του γράφου με σημασιολογία για την εξέλιξη του σχήματος της βάσης, περιλαμβάνει κλάσεις που υλοποιούν ένα γράφο, ο οποίος προσφέρει τη δυνατότητα ορισμού πιθανών γεγονότων, πολιτικών χειρισμού αυτών και εύρεση της επίδρασής τους στο σύστημα. Κάθε κλάση του υποσυστήματος HecataeusEvolutionManager επεκτείνει την αντίστοιχη κλάση του υποσυστήματος HecataeusGraph. Το υποσύστημα HecataeusJungGraph, το οποίο αντιστοιχεί στο τμήμα για την οπτικοποίηση του γράφου, περιλαμβάνει κλάσεις που υποστηρίζουν την οπτική αναπαράσταση του γράφου και χρησιμοποιούν τις αντίστοιχες κλάσεις του HecataeusEvolutionGraph. Τέλος, το υποσύστημα Hecataeus, το οποίο αντιστοιχεί στο τμήμα για τη διεπαφή της εφαρμογής, στο προσφέρει το γραφικό περιβάλλον της εφαρμογής. Τα υποσυστήματα του συστήματος και ο τρόπος που αυτά επικοινωνούν μεταξύ τους απεικονίζονται στην Εικόνα 4.1.



Εικόνα 4.1. Διάγραμμα UML με τα υποσυστήματα του συστήματος και τον τρόπο που αυτά επικοινωνούν μεταξύ τους.

Κλάσεις του πακέτου HecataeusGraph

Το πακέτο περιλαμβάνει τις κλάσεις που υλοποιούν ένα γράφο που αναπαριστά μια βάση δεδομένων και τα ερωτήματα προς αυτή. Ο γράφος αποτελείται από ένα σύνολο κόμβων και ένα ακμών. Οι κόμβοι και οι ακμές παίρνουν τον τύπο τους από ένα προκαθορισμένο σύνολο με τους επιτρεπτούς τύπους, αντίστοιχα. Το διάγραμμα των κλάσεων του πακέτου απεικονίζεται στην Εικόνα 4.2.

4.1.1 Η κλάση HecataeusEdgeType

Η κλάση συγκεντρώνει τους εννέα δυνατούς τύπους που μπορεί να πάρει μια ακμή γράφου που αναπαριστά το σχήμα μιας βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν και παρέχει τη δυνατότητα για αναφορά σε έναν τύπο είτε με μια συμβολοσειρά (που είναι πιο ευκολομνημόνευτη), είτε με έναν ακέραιο (που βοηθά στο να διατρέχουμε όλους τους τύπους ακμών). Η κλάση παρέχει επίσης τρόπους μετατροπής από τη μια αναπαράσταση του τύπου στην άλλη.

4.1.2 Η κλάση HecataeusEdge

Όπως δηλώνεται και από το όνομά της, η κλάση υλοποιεί μια κατευθυνόμενη ακμή ενός γράφου, ο οποίος μπορεί να χρησιμοποιηθεί για την αναπαράσταση του σχήματος μιας βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν. Η ακμή έχει ως ιδιότητες το όνομά της, τον τύπο της (έναν από τους εννέα τύπους που δηλώνονται στην κλάση HecataeusEdgeType), τον κόμβο πηγή και τον κόμβο προορισμό, καθώς και το κλειδί της, έναν αριθμό που την προσδιορίζει μονοσήμαντα.

4.1.3 Η κλάση HecataeusEdges

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusEdge, δηλαδή ένα σύνολο από κατευθυνόμενες ακμές γράφου που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν και παρέχει συναρτήσεις για το χειρισμό τους (εισαγωγή, εξαγωγή, αναζήτηση κ.λ.π.)

4.1.4 Η κλάση HecataeusNodeType

Η κλάση συγκεντρώνει τους δώδεκα δυνατούς τύπους που μπορεί να πάρει ένας κόμβος γράφου που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν και παρέχει τη δυνατότητα για αναφορά σε έναν τύπο είτε με μια συμβολοσειρά, είτε με έναν ακέραιο. Η κλάση παρέχει επίσης τρόπους μετατροπής από τη μια αναπαράσταση του τύπου στην άλλη.

4.1.5 Η κλάση HecataeusNode

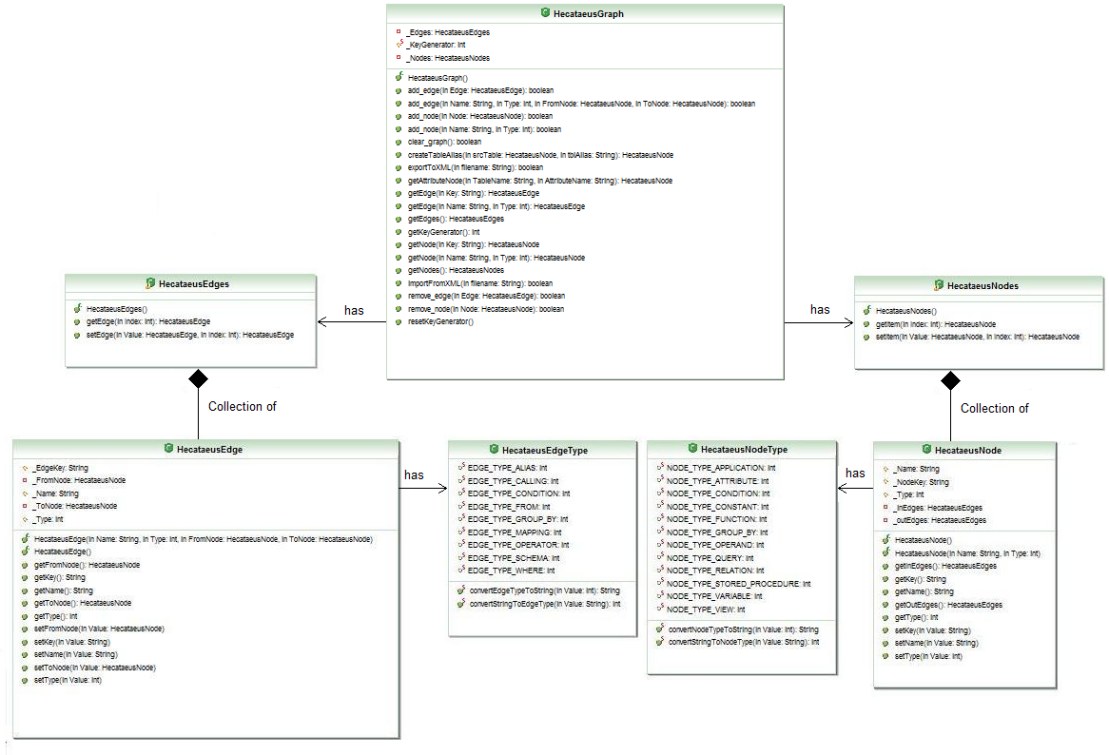
Η κλάση αποτελεί την υλοποίηση ενός κόμβου γράφου για την αναπαράσταση του σχήματος μιας βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν. Ο κόμβος έχει ως ιδιότητες το όνομά του, τον τύπο του (έναν από τους δώδεκα τύπους που δηλώνονται στην κλάση HecataeusNodeType) καθώς και το κλειδί του, έναν αριθμό που τον προσδιορίζει μονοσήμαντα.

4.1.6 Η κλάση HecataeusNodes

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusNode, δηλαδή ένα σύνολο από κόμβους γράφου για την αναπαράσταση του σχήματος μιας βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν και παρέχει συναρτήσεις για τον χειρισμό τους.

4.1.7 Η κλάση HecataeusGraph

Η κλάση υλοποιεί έναν κατευθυνόμενο γράφο που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν. Ο γράφος αποτελείται από ένα σύνολο κόμβων και ένα σύνολο ακμών. Η κλάση περιέχει μεθόδους για τη δημιουργία και την τροποποίηση του γράφου, καθώς και την προσπέλαση των στοιχείων του. Όταν προστίθεται ένας νέος κόμβος ή μια νέα ακμή στο γράφο, (δηλαδή σε ένα αντικείμενο αυτής της κλάσης), ο γράφος φροντίζει να δίνει σε κάθε στοιχείο μια μοναδική τιμή για το κλειδί του.



Εικόνα 4.2. Διάγραμμα κλάσεων για το υποσύστημα-πακέτο HecataeusGraph.

Κλάσεις του πακέτου HecataeusEvolutionManager

Οι κλάσεις του υποσυστήματος επεκτείνουν τις αντίστοιχες κλάσεις του πακέτου HecataeusGraph και προσφέρουν επιπλέον δυνατότητες όσον αφορά στην ανάλυση της εξέλιξης του σχήματος της βάσης δεδομένων που αναπαριστά ο γράφος. Το διάγραμμα των κλάσεων του πακέτου απεικονίζεται στην Εικόνα 4.3.

4.1.8 Η κλάση EvolutionEventType

Η κλάση συγκεντρώνει τους δέκα τύπους γεγονότων που μπορούν να συμβούν στο σχήμα μιας βάσης δεδομένων και παρέχει τη δυνατότητα για αναφορά σε έναν τύπο είτε με μια συμβολοσειρά, είτε με έναν ακέραιο. Η κλάση παρέχει επίσης τρόπους μετατροπής από τη μια αναπαράσταση του τύπου στην άλλη.

4.1.9 Η κλάση EvolutionPolicyType

Η κλάση συγκεντρώνει τους τρεις τύπους πολιτικών που ορίστηκαν σε προηγούμενη ενότητα και παρέχει τη δυνατότητα για αναφορά σε έναν τύπο είτε με μια συμβολοσειρά, είτε με έναν ακέραιο. Η κλάση παρέχει επίσης τρόπους μετατροπής από τη μια αναπαράσταση του τύπου στην άλλη.

4.1.10 Η κλάση EvolutionStatus

Η κλάση συγκεντρώνει τους τρεις τύπους κατάστασης στην οποία μπορεί να βρεθεί ένας κόμβος κατά την εξέλιξη της βάσης μετά τον ορισμό ενός γεγονότος. Η κατάσταση ενός κόμβου μπορεί να είναι μια από τις παρακάτω:

- Επηρεασμένος από το γεγονός.
- Προς διαγραφή.
- Προς διαγραφή κόμβου-παιδιού.
- Προς προσθήκη κόμβου-παιδιού.

ή τίποτα αν δεν επηρεάζεται καθόλου από το γεγονός.

Μια ακμή, επίσης, μπορεί να βρεθεί στους δύο από τους παραπάνω τύπους:

- Επηρεασμένη από το γεγονός.
- Προς διαγραφή.

ή τίποτα αν δεν επηρεάζεται καθόλου από το γεγονός.

Η κλάση παρέχει τη δυνατότητα για αναφορά σε έναν τύπο είτε με μια συμβολοσειρά, είτε με έναν ακέραιο, καθώς και τρόπους μετατροπής από τη μια αναπαράσταση του τύπου στην άλλη.

4.1.11 Η κλάση EvolutionMessage

Η κλάση αναπαριστά ένα μήνυμα, που στέλνει ένας κόμβος στους κόμβους που συνδέονται με αυτόν με εισερχόμενες ακμές και χρησιμοποιείται από τη συνάρτηση που βρίσκει τα τμήματα του γράφου που επηρεάζονται από ένα γεγονός και θέτει ανάλογα την κατάστασή τους. Ένα μήνυμα αποτελείται από το μοναδικό αριθμό που χαρακτηρίζει τη συγκεκριμένη διαδικασία εύρεσης της επίδρασης ενός συγκεκριμένου γεγονότος, τον κόμβο που στέλνει το μήνυμα, τον κόμβο που λαμβάνει το μήνυμα, το γεγονός για το οποίο κλήθηκε η διαδικασία, τον τύπο της ακμής που συνδέει τον κόμβο αποστολέα με τον κόμβο παραλήπτη και την πολιτική χειρισμού του γεγονότος του κόμβου που έστειλε το μήνυμα.

4.1.12 Η κλάση HecataeusEvolutionNode

Η κλάση υλοποιεί έναν κόμβο γράφου που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, όπως και η κλάση HecataeusNode από την οποία κληρονομεί. Η επιπλέον δυνατότητα που προσφέρει είναι ο ορισμός μιας συλλογής πολιτικών και γεγονότων για αυτόν το κόμβο και η προσπέλασή τους. Επίσης, ο κόμβος έχει κατάσταση, η οποία αλλάζει για να δείξει τον τρόπο που μια αλλαγή στο σχήμα της βάσης επιδρά πάνω στον κόμβο αυτό.

4.1.13 Η κλάση HecataeusEvolutionNodes

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusEvolutionNodes, δηλαδή ένα σύνολο από κόμβους γράφου, που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, με δυνατότητα ορισμού πολιτικών και γεγονότων και παρέχει συναρτήσεις για τον χειρισμό τους.

4.1.14 Η κλάση HecataeusEvolutionEdge

Η κλάση υλοποιεί μια κατευθυνόμενη ακμή γράφου που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, όπως και η κλάση HecataeusEdge από την οποία κληρονομεί. Η μόνη διαφορά είναι ότι οι κόμβοι που αποτελούν τα άκρα της πλευράς είναι αντικείμενα τύπου HecataeusEvolutionNode, δηλαδή μπορεί να οριστούν σ' αυτούς πολιτικές και γεγονότα και ότι η ακμή έχει κατάσταση, η οποία αλλάζει για να δείξει τον τρόπο που μια αλλαγή στο σχήμα της βάσης επιδρά πάνω στην ακμή αυτή.

4.1.15 Η κλάση HecataeusEvolutionEdges

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusEvolutionEdges, δηλαδή ένα σύνολο από κατευθυνόμενες ακμές γράφου, που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, και παρέχει συναρτήσεις για το χειρισμό τους.

4.1.16 Η κλάση HecataeusEvolutionPolicy

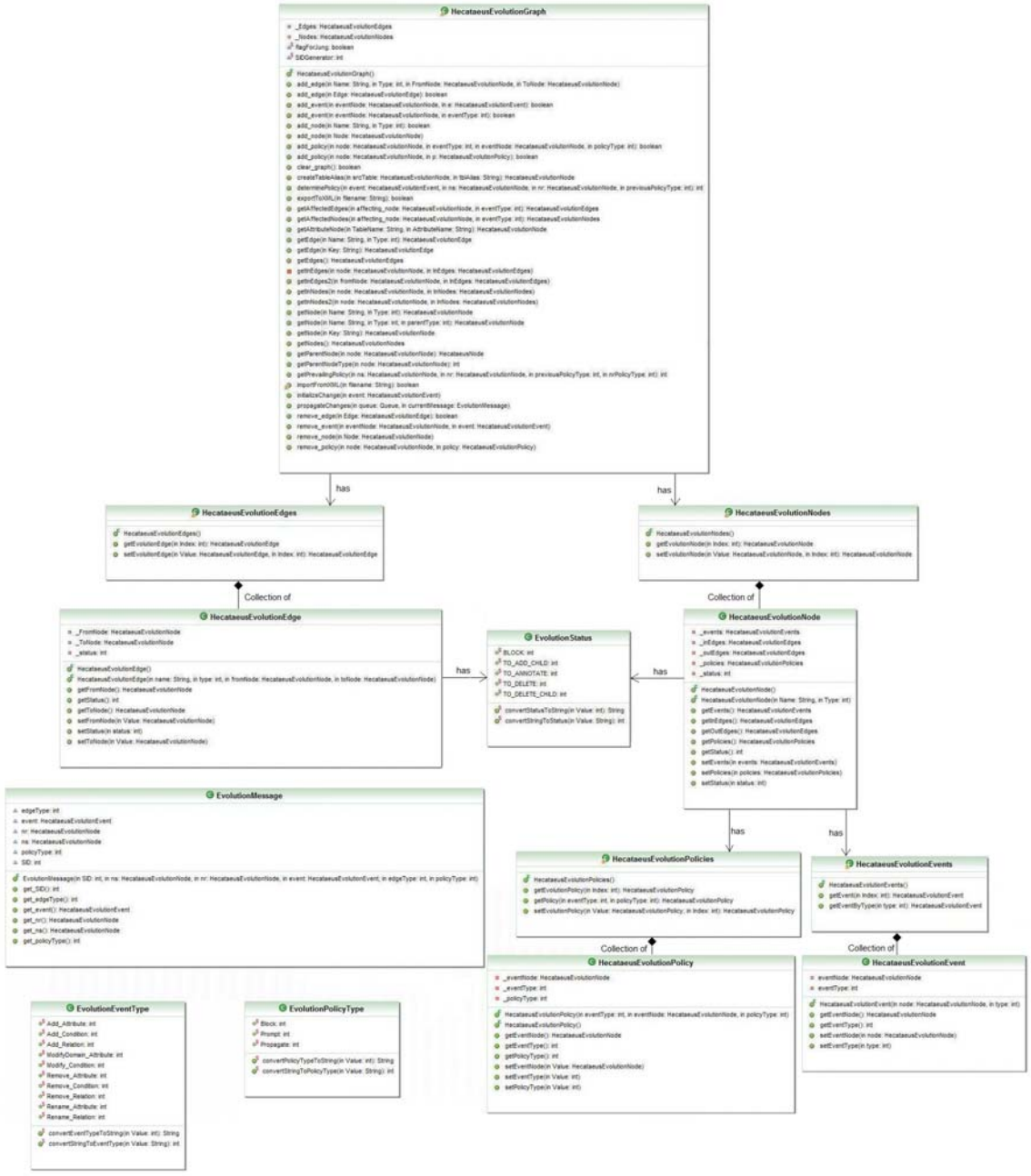
Η κλάση αναπαριστά μια πολιτική. Ένα αντικείμενο αυτής της κλάσης έχει ως ιδιότητες τον τύπο της πολιτικής, τον τύπο του γεγονότος τον οποίο αυτή η πολιτική αντιμετωπίζει και τον κόμβο τον οποίο αφορά αυτό το γεγονός.

4.1.17 Η κλάση HecataeusEvolutionPolicies

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusEvolutionPolicy, δηλαδή ένα σύνολο από πολιτικές και παρέχει συναρτήσεις για τον χειρισμό τους.

4.1.18 Η κλάση HecataeusEvolutionEvent

Η κλάση αναπαριστά ένα γεγονός που μπορεί να λάβει χώρα σε ένα κόμβο του γράφου. Ένα αντικείμενο αυτής της κλάσης έχει ως ιδιότητες τον τύπο του γεγονότος και τον κόμβο στον οποίο ορίστηκε το γεγονός.



Εικόνα 4.3. Διάγραμμα κλάσεων για το υποσύστημα-πακέτο HecataeusEvolutionManager.

4.1.19 Η κλάση HecataeusEvolutionEvents

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusEvolutionEvent, δηλαδή ένα σύνολο από γεγονότα και παρέχει συναρτήσεις για το χειρισμό τους.

4.1.20 Η κλάση HecataeusEvolutionGraph

Η κλάση αναπαριστά έναν κατευθυνόμενο γράφο που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, όπως και η κλάση HecataeusGraph από την οποία κληρονομεί. Η επιπλέον δυνατότητες που προσφέρει αυτή η κλάση αφορούν στην

εξέλιξη του σχήματος μιας βάσης δεδομένων. Συγκεκριμένα, προσφέρει τη δυνατότητα ορισμού πολιτικών στους κόμβους του γράφου, τη δυνατότητα διαγραφής πολιτικών από τους κόμβους του γράφου, τη δυνατότητα εύρεσης των κόμβων και των πλευρών που επηρεάζονται από ένα γεγονός που λαμβάνει χώρα στο σχήμα της βάσης δεδομένων και τη δυνατότητα εύρεσης της επικρατούσας πολιτικής σε περίπτωση που έχουμε σύγκρουση πολιτικών.

Κλάσεις του πακέτου HecataeusJungGraph

Το πακέτο υλοποιεί ένα γράφο, ο οποίος μπορεί επιπλέον να οπτικοποιηθεί. Το διάγραμμα των κλάσεων του πακέτου απεικονίζεται στην Εικόνα 4.4.

4.1.21 Η κλάση HecataeusJungEdge

Η κλάση υλοποιεί μια κατευθυνόμενη ακμή γράφου όπως και η κλάση HecataeusEvolutionEdge. Η διαφορά είναι ότι αυτή η ακμή παρέχει τη δυνατότητα οπτικοποίησής της. Η κλάση έχει μεταβλητές που καθορίζουν το πώς θα φαίνεται η ακμή όταν οπτικοποιηθεί, ανάλογα με την τρέχουσα κατάσταση. Για παράδειγμα, όταν η ακμή είναι επιλεγμένη στο πλάνο του γράφου, τότε το χρώμα της θα είναι κυανό.

Κάθε αντικείμενο της κλάσης HecataeusJungEdge συνδέεται με ένα αντικείμενο της κλάσης HecataeusEvolutionEdge. Έτσι, όσον αφορά στην εξέλιξη του σχήματος της βάσης δεδομένων, αυτή η κλάση **δεν** παρέχει συναρτήσεις, αλλά όταν χρειαστεί να εκτελεστεί μια τέτοια λειτουργία, όπως για παράδειγμα η εύρεση των πλευρών που επηρεάζονται από ένα γεγονός, καλείται η αντίστοιχη λειτουργία με αντικείμενα HecataeusEvolutionEdge.

4.1.22 Η κλάση HecataeusJungEdges

Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusJungEdges, δηλαδή ένα σύνολο από ακμές γράφου με δυνατότητα οπτικοποίησης και παρέχει συναρτήσεις για το χειρισμό τους.

4.1.23 Η κλάση HecataeusJungNode

Η κλάση υλοποιεί έναν κόμβο γράφου, όπως και η κλάση HecataeusEvolutionNode. Η διαφορά είναι, όπως και με τις ακμές, ότι ο κόμβος αυτός έχει τη δυνατότητα οπτικοποίησής του. Όπως και η κλάση HecataeusJungEdge, η κλάση HecataeusJungNode έχει μεταβλητές που καθορίζουν το πώς θα φαίνεται ο κόμβος όταν οπτικοποιηθεί, ανάλογα με την τρέχουσα κατάσταση.

Ομοίως με την κλάση HecataeusJungEdge, κάθε αντικείμενο της κλάσης HecataeusJungNode συνδέεται με ένα αντικείμενο της κλάσης HecataeusEvolutionNode.

Έτσι, όσον αφορά στην εξέλιξη του πληροφοριακού συστήματος, αυτή η κλάση **δεν** παρέχει συναρτήσεις, αλλά όταν χρειαστεί να εκτελεστεί μια τέτοια λειτουργία, όπως για παράδειγμα ο ορισμός μιας πολιτικής σ' ένα αντικείμενο της κλάσης HecataeusJungNode, καλείται η αντίστοιχη λειτουργία με το αντικείμενο της κλάσης HecataeusEvolutionNode.

4.1.24 Η κλάση HecataeusJungNodes

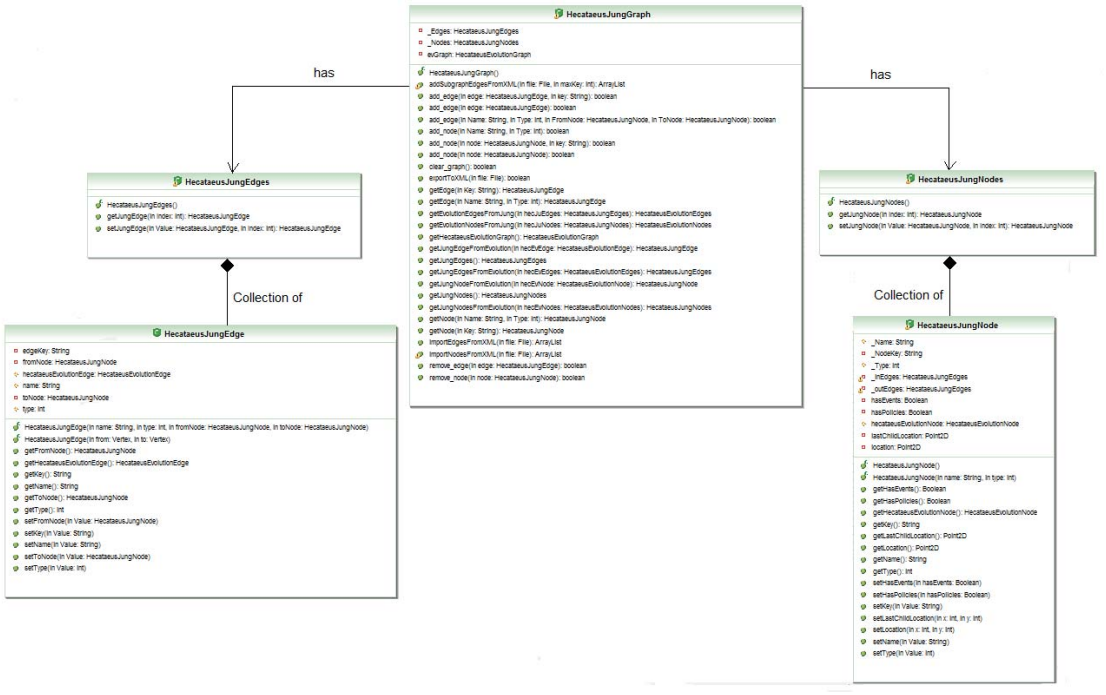
Η κλάση είναι μια συλλογή από αντικείμενα τύπου HecataeusJungNodes, δηλαδή ένα σύνολο από κόμβους γράφου, που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, με δυνατότητα οπτικοποίησης και παρέχει συναρτήσεις για το χειρισμό τους.

4.1.25 Η κλάση HecataeusJungGraph

Η κλάση υλοποιεί έναν κατευθυνόμενο γράφο που αναπαριστά μια βάση δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν, όπως και η κλάση HecataeusEvolutionManager, με επιπλέον δυνατότητα οπτικοποίησης.

Κάθε αντικείμενο της κλάσης HecataeusJungGraph συνδέεται με ένα αντικείμενο της κλάσης HecataeusEvolutionGraph.

Όσον αφορά στην εξέλιξη του πληροφοριακού συστήματος, αυτή η κλάση **δεν** παρέχει συναρτήσεις, αλλά όταν χρειαστεί να εκτελεστεί μια τέτοια λειτουργία, όπως για παράδειγμα η εύρεση της επικρατούσας πολιτικής, καλείται η αντίστοιχη λειτουργία του HecataeusEvolutionGraph.



Εικόνα 4.4. Διάγραμμα κλάσεων για το υποσύστημα-πακέτο HecataeusJungGraph.

Κλάσεις του πακέτου Hecataeus

Οι κλάσεις του πακέτου αυτού υλοποιούν το γραφικό περιβάλλον της εφαρμογής, όπου γίνεται η οπτικοποίηση του γράφου και ο γραφικός χειρισμός του. Το διάγραμμα των κλάσεων του πακέτου απεικονίζεται στην Εικόνα 4.5.

4.1.26 Η κλάση HecataeusModalGraphMouse

Είναι η κλάση που παρέχει τις δύο καταστάσεις λειτουργίας του ποντικιού, όπως αυτές περιγράφηκαν στην παράγραφο 3.2.4. και την εναλλαγή από τη μια κατάσταση στην άλλη.

4.1.27 Η κλάση JungEditingPopupGraphMousePlugin

Η κλάση περιλαμβάνει επιπλέον λειτουργικότητα που παρέχεται με αριστερό ή δεξί πάτημα του ποντικιού.

Αν το δεξί πάτημα του ποντικιού έγινε πάνω σε κόμβο, τότε:

- Όταν το πλήθος των επιλεγμένων κόμβων είναι μεγαλύτερο του μηδενός εμφανίζεται η επιλογή για δημιουργία νέας ακμής από τον/τους επιλεγμένους κόμβους στον κόμβο στον οποίο πατήθηκε το ποντίκι (εκτός αν ο επιλεγμένος κόμβος ταυτίζεται με τον κόμβο στον οποίο πατήθηκε το ποντίκι: δεν επιτρέπονται οι αυτοβρόχοι).
- Όταν το πλήθος των επιλεγμένων κόμβων είναι μεγαλύτερο της μονάδας εμφανίζεται η επιλογή για διαγραφή των επιλεγμένων κόμβων.
- Όταν το πλήθος των επιλεγμένων κόμβων είναι μεγαλύτερο της μονάδας εμφανίζεται η επιλογή για ορισμό μιας πολιτικής στους επιλεγμένους κόμβους (βλ. κλάση 4.1.27.2 - AddPolicy).
- Εμφανίζεται η επιλογή για διαγραφή του κόμβου.
- Εμφανίζεται η επιλογή για εμφάνιση του παραθύρου για εμφάνιση ή/και αλλαγή του ονόματος και του τύπου του κόμβου (βλ. κλάση 4.1.27.3 - NameType).
- Εμφανίζεται η επιλογή για εμφάνιση του παραθύρου για χειρισμό πολιτικών πάνω στον κόμβο (βλ. κλάση 4.1.27.4 - Policies).
- Εμφανίζεται η επιλογή για εμφάνιση του παραθύρου χειρισμού γεγονότων πάνω στον κόμβο (βλ. κλάση 4.1.27.5 - Events).

Αν το δεξί πάτημα του ποντικιού έγινε πάνω σε ακμή, τότε:

- Εμφανίζεται η επιλογή για διαγραφή της ακμής.
- Εμφανίζεται η επιλογή για εμφάνιση ή/και αλλαγή του ονόματος και του τύπου της ακμής.
- Αν το δεξί πάτημα του ποντικιού έγινε οπουδήποτε στον κενό χώρο (όχι πάνω σε κόμβο ή ακμή), τότε:

- Εμφανίζεται η επιλογή για δημιουργία νέου κόμβου.

Οι παρακάτω κλάσεις ορίζονται μέσα στην κλάση JungEditingPopupGraphMousePlugin και χρειάζονται για την εμφάνιση των παραθύρων

4.1.27.1 Η κλάση ConstraintsBuild

Η κλάση χρησιμοποιείται για τον καθορισμό ορισμένων παραμέτρων του GridBagLayout.

4.1.27.2 Η κλάση AddPolicy

Η κλάση υλοποιεί ένα παράθυρο, στο οποίο ο χρήστης μπορεί να ορίσει μια καινούρια πολιτική για τους επιλεγμένους κόμβους στο πλάνο.

4.1.27.3 Η κλάση NameType

Η κλάση υλοποιεί ένα παράθυρο, στο οποίο ο χρήστης καθορίζει το όνομα και τον τύπο του κόμβου ή της ακμής.

4.1.27.4 Η κλάση Policies

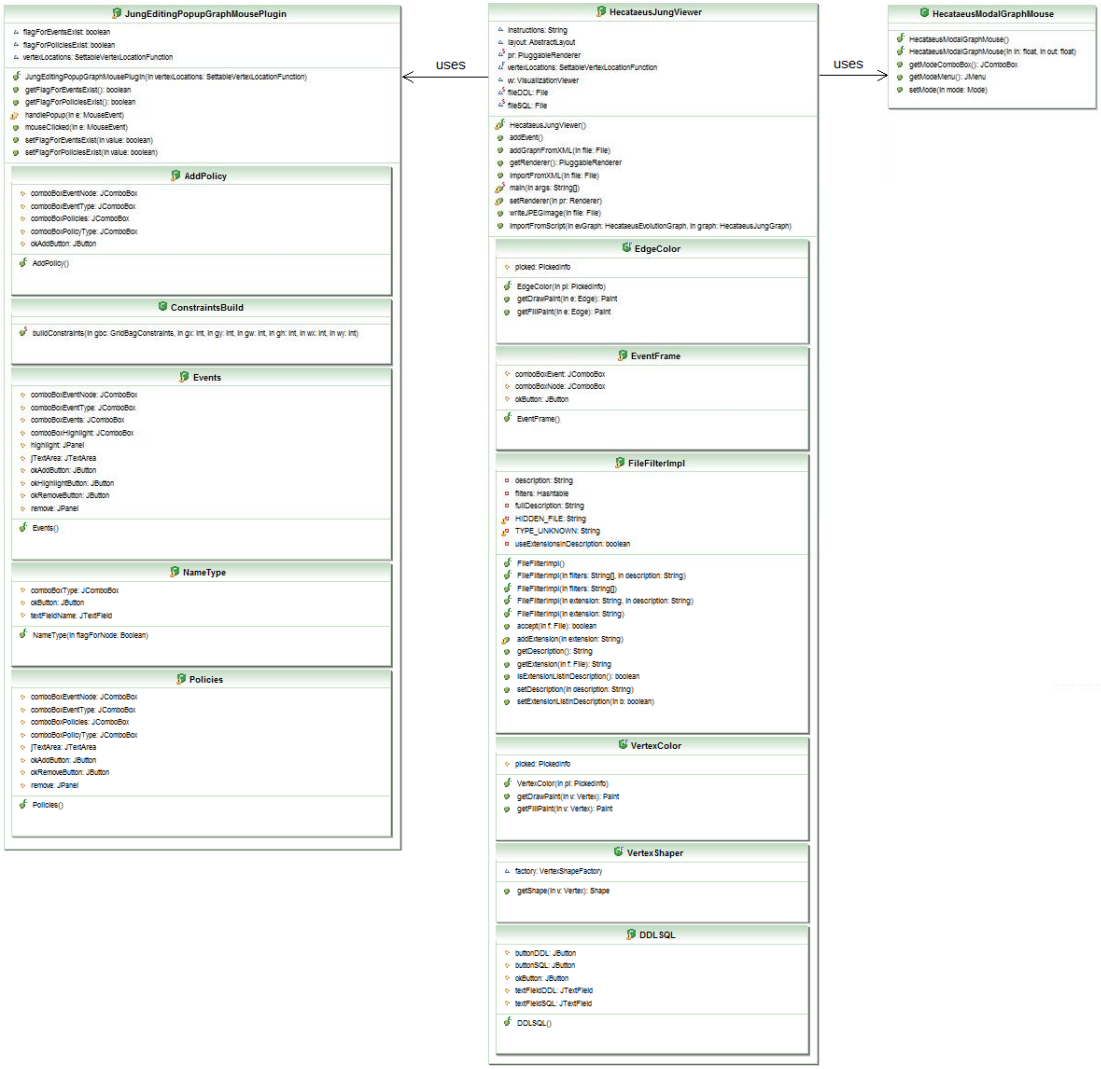
Η κλάση υλοποιεί ένα παράθυρο με καρτέλες στο οποίο παρέχονται επιλογές για:

- Την εμφάνιση όλων των πολιτικών που έχουν οριστεί για τον κόμβο.
- Τον ορισμό καινούριας πολιτικής για τον κόμβο.
- Την αφαίρεση μιας πολιτικής από τη συλλογή πολιτικών του κόμβου.

4.1.27.5 Η κλάση Events

Η κλάση υλοποιεί ένα παράθυρο με καρτέλες στο οποίο παρέχονται επιλογές για:

- Την εμφάνιση όλων των γεγονότων που έχουν οριστεί για τον κόμβο.
- Το κατάλληλο χρωματικό τονισμό των κόμβων και των ακμών του γράφου που επηρεάζονται από ένα γεγονός.
- Τον ορισμό καινούριου γεγονότος για τον κόμβο.
- Την αφαίρεση μιας πολιτικής από τη συλλογή πολιτικών του κόμβου.



Εικόνα 4.5. Διάγραμμα κλάσεων για το υποσύστημα-πακέτο Hecataeus.

4.1.28 Η κλάση HecataeusJungViewer

Είναι η κύρια κλάση που παρέχει το γραφικό περιβάλλον της εφαρμογής. Τρέχοντας αυτή την κλάση εμφανίζεται στην οθόνη του χρήστη ένα παράθυρο που περιέχει το πλάνο του γράφου και ένα μενού με τις λειτουργίες που μπορεί να εφαρμόσει ο χρήστης.

Οι παρακάτω κλάσεις ορίζονται μέσα στην κλάση HecataeusJungViewer.

4.1.28.1 Η κλάση FileFilterImpl

Η κλάση υλοποιεί είναι ένα φίλτρο που επιτρέπει μόνο σε αρχεία συγκεκριμένου τύπου να εμφανιστούν στο παράθυρο επιλογής αρχείων.

4.1.28.2 Η κλάση EventFrame

Η κλάση υλοποιεί ένα παράθυρο για τον ορισμό νέου γεγονότος πάνω σε ένα κόμβο.

4.1.28.3 Η κλάση VertexShaper

Η κλάση καθορίζει το σχήμα των κόμβων του γράφου, ανάλογα με τον τύπο τους. Συγκεκριμένα:

- ο κόμβος τύπου «ερώτημα» αναπαριστάται με εξάγωνο
- ο κόμβος τύπου «σχέση» αναπαριστάται με κύκλο
- ο κόμβος τύπου «όψη» αναπαριστάται με τρίγωνο
- ο κόμβος οποιουδήποτε άλλου τύπου αναπαριστάται με τετράγωνο.

(Η κλάση προσφέρεται ως επιπλέον δυνατότητα, η οποία δε χρησιμοποιήθηκε στην εφαρμογή.)

4.1.28.4 Η κλάση VertexColor

Η κλάση είναι υπεύθυνη για τον κατάλληλο χρωματισμό των κόμβων του γράφου. Συγκεκριμένα, ανάλογα με τον τύπο του, ένας κόμβος χρωματίζεται ως εξής:

- ο κόμβος τύπου «ερώτημα» χρωματίζεται με μπλε.
- ο κόμβος τύπου «σχέση» χρωματίζεται με μπλε σκούρο.
- ο κόμβος τύπου «περιορισμός» χρωματίζεται με μπλε ανοιχτό.
- ο κόμβος τύπου «όψη» χρωματίζεται με θαλασσί.
- ο κόμβος τύπου «γνώρισμα» χρωματίζεται με γκρι.
- ο κόμβος οποιουδήποτε άλλου τύπου χρωματίζεται με λευκό.

Το χρώμα του γράφου αλλάζει στις παρακάτω περιπτώσεις:

- όταν ένας κόμβος είναι επιλεγμένος το χρώμα του είναι κυανό.
- όταν έχει πολιτική το χρώμα του είναι κίτρινο.
- όταν έχει γεγονός το χρώμα του είναι ροζ.

Επίσης, οι κόμβοι του μετασχηματισμένου γράφου που προκύπτει από την εξέλιξη της βάσης ως αποτέλεσμα ενός γεγονότος χρωματίζονται ως εξής:

- κόκκινοι αν πρέπει να διαγραφούν.
- Σκούροι κόκκινοι αν πρέπει να διαγραφεί ένας κόμβος-παιδί τους.
- πράσινοι αν πρέπει να προστεθούν σ’ αυτούς κόμβοι-παιδιά.
- μοβ αν επηρεάζονται με κάθε άλλο τρόπο από το γεγονός

- μαύροι αν εμποδίζουν την προώθηση του γεγονότος.

Σε κάθε άλλη περίπτωση το χρώμα των κόμβων είναι ανοιχτό γκρι.

4.1.28.5 Η κλάση EdgeColor

Η κλάση είναι υπεύθυνη για τον κατάλληλο τονισμό των ακμών του γράφου. Συγκεκριμένα,

- όταν μια ακμή είναι επιλεγμένη τονίζεται με κυανό.

Επίσης, οι ακμές του μετασχηματισμένου γράφου που προκύπτει από την εξέλιξη της βάσης ως αποτέλεσμα ενός γεγονότος τονίζονται ως εξής:

- κόκκινες αν πρέπει να διαγραφούν
- μοβ αν επηρεάζονται με κάθε άλλο τρόπο από το γεγονός

Σε κάθε άλλη περίπτωση οι ακμές δεν είναι τονισμένες (λευκό).

4.1.28.6 Η κλάση DDLSQL

Η κλάση υλοποιεί ένα παράθυρο για εισαγωγή γράφου απευθείας (χωρίς τη μετατροπή σε μορφή xml) από αρχεία τύπου sql. Συγκεκριμένα, εισάγεται το αρχείο που περιέχει τον ορισμό των δεδομένων (DDL) και το αρχείο που περιέχει τις όψεις και τα ερωτήματα (SQL). Ο γράφος που εισάγεται οπτικοποιείται στο πλάνο.

4.2 Περιγραφή Κλάσεων

Σ' αυτήν την παράγραφο θα περιγραφούν συνοπτικά οι λειτουργίες/μεθόδοι/συναρτήσεις κάθε κλάσης.

Κλάσεις του πακέτου HecataeusGraph

4.2.1 Λειτουργίες της κλάσης HecataeusEdge

- **public** `HecataeusEdge()`: Δημιουργία καινούργιου αντικειμένου ακμής.
- **public** `HecataeusEdge(String Name, int Type, HecataeusNode FromNode, HecataeusNode ToNode)`: Δημιουργία καινούργιου αντικειμένου ακμής με όνομα Name, τύπο Type, κόμβο πηγή `FromNode` και κόμβο προορισμού `ToNode`.
- **public void** `setName(String Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «όνομα» της ακμής.
- **public void** `setType(int Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «τύπος» της ακμής.

- **public void** `setKey(String Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κλειδί» της ακμής.
- **public void** `setFromNode(HecataeusNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος πηγή» της ακμής.
- **public void** `setToNode(HecataeusNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος προορισμού» της ακμής.
- **public** `String getName()`: Εύρεση της τιμής της ιδιότητας «όνομα» της ακμής.
- **public int** `getType()`: Εύρεση της τιμής της ιδιότητας «τύπος» της ακμής.
- **public** `String getKey()`: Εύρεση της τιμής της ιδιότητας «κλειδί» της ακμής.
- **public** `HecataeusNode getFromNode()`: Εύρεση της τιμής της ιδιότητας «κόμβος πηγή» της ακμής.
- **public** `HecataeusNode getToNode()`: Εύρεση της τιμής της ιδιότητας «κόμβος προορισμού» της ακμής.

4.2.2 Λειτουργίες της κλάσης HecataeusEdges

- **public** `HecataeusEdges()`: Δημιουργία καινούργιου αντικειμένου λίστας ακμών.
- **public** `HecataeusEdge getEdge(int Index)`: Εύρεση της ακμής στη θέση `Index` στη λίστα ακμών.
- **public** `HecataeusEdge setEdge(HecataeusEdge Value, int Index)`: Εισαγωγή της ακμής `Value` στη θέση `Index` στη λίστα ακμών.

4.2.3 Λειτουργίες της κλάσης HecataeusEdgeType

- **public static** `String convertEdgeTypeToString(int Value)` : Μετατροπή από την αναπαράσταση του τύπου ακμής με ακέραιο `Value` στην αναπαράσταση τύπου με συμβολοσειρά.
- **public static int** `convertStringToEdgeType(String Value)` : Μετατροπή από την αναπαράσταση του τύπου ακμής με συμβολοσειρά `Value` στην αναπαράσταση τύπου με ακέραιο.

4.2.4 Λειτουργίες της κλάσης HecataeusNode

- **public** `HecataeusNode()` : Δημιουργία καινούργιου αντικειμένου κόμβου.
- **public** `HecataeusNode(String Name, int Type)` : Δημιουργία καινούργιου αντικειμένου κόμβου με όνομα `Name` και τύπο `Type`.

- **public void** `setName(String Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «όνομα» του κόμβου.
- **public void** `setType(int Value)` : Ανάθεση της τιμής `Value` στην ιδιότητα «τύπος» του κόμβου.
- **public void** `setKey(String Value)` : Ανάθεση της τιμής `Value` στην ιδιότητα «κλειδί» του κόμβου.
- **public** `String getName()`: Εύρεση της τιμής της ιδιότητας «όνομα» του κόμβου.
- **public int** `getType()` : Εύρεση της τιμής της ιδιότητας «τύπος» του κόμβου.
- **public** `String getKey()` : Εύρεση της τιμής της ιδιότητας «κλειδί» του κόμβου.
- **public** `HecataeusEdges getOutEdges()` : Εύρεση των ακμών που αναχωρούν από τον κόμβο.
- **public** `HecataeusEdges getInEdges()` : Εύρεση των ακμών που προσπίπτουν στον κόμβο.

4.2.5 Λειτουργίες της κλάσης HecataeusNodes

- **public** `HecataeusNodes()` : Δημιουργία καινούργιου αντικειμένου λίστας κόμβων.
- **public** `HecataeusNode getItem(int Index)` : Εύρεση του κόμβου στη θέση `Index` στη λίστα κόμβων.
- **public** `HecataeusNode setItem(HecataeusNode Value, int Index)` : Εισαγωγή του κόμβου `Value` στη θέση `Index` στη λίστα κόμβων.

4.2.6 Λειτουργίες της κλάσης HecataeusNodeType

- **public static** `String convertNodeTypeToString(int Value)` : Μετατροπή από την αναπαράσταση του τύπου κόμβου με ακέραιο `Value` στην αναπαράσταση τύπου με συμβολοσειρά.
- **public static int** `convertStringToNodeType(String Value)`: Μετατροπή από την αναπαράσταση του τύπου κόμβου με συμβολοσειρά `Value` στην αναπαράσταση τύπου με ακέραιο.

4.2.7 Λειτουργίες της κλάσης HecataeusGraph

- **public** `HecataeusGraph()`: Δημιουργία καινούριου αντικειμένου γράφου.
- **public boolean** `add_node(String Name, int Type)`: Εισαγωγή καινούριου κόμβου στο γράφο με όνομα `Name` και τύπο `Type`.

- **public boolean** `add_node(HecataeusNode Node)` : Εισαγωγή του κόμβου `Node` στο γράφο.
- **public boolean** `remove_node(HecataeusNode Node)` : Αφαίρεση του κόμβου `Node` από το γράφο.
- **public boolean** `add_edge(String Name, int Type, HecataeusNode FromNode, HecataeusNode ToNode)` : Εισαγωγή καινούριας ακμής στο γράφο με όνομα `Name`, τύπο `Type`, κόμβο πηγή `FromNode` και κόμβο προορισμού `ToNode`.
- **public boolean** `add_edge(HecataeusEdge Edge)` : Εισαγωγή της ακμής `Edge` στο γράφο.
- **public boolean** `remove_edge(HecataeusEdge Edge)` : Αφαίρεση της ακμής `Edge` από το γράφο.
- **public boolean** `clear_graph()` : Διαγραφή του γράφου.
- **public** `HecataeusNode createTableAlias(HecataeusNode srcTable, String tblAlias)`: Δημιουργία ψευδωνύμου της σχέσης `srcTable` με όνομα `tblAlias`.
- **public** `HecataeusNode getNode(String Key)` : Εύρεση του κόμβου του γράφου με τιμή κλειδιού `Key`.
- **public** `HecataeusNode getNode(String Name, int Type)`: Εύρεση του κόμβου του γράφου με όνομα `Name` και τύπο `Type`.
- **public** `HecataeusEdge getEdge(String Key)`: Εύρεση της ακμής του γράφου με τιμή κλειδιού `Key`.
- **public** `HecataeusEdge getEdge(String Name, int Type)`: Εύρεση της ακμής του γράφου με όνομα `Name` και τύπο `Type`.
- **public** `HecataeusNode getAttributeNode(String TableName, String AttributeName)`: Εύρεση ενός κόμβου τύπου «γνώρισμα» από το όνομα του `AttributeName` και το όνομα του κόμβου γονέα `TableName` (ο οποίος μπορεί να είναι τύπου «σχέση», «ερώτημα» ή «όψη»).
- **public boolean** `importFromXML(String filename)`: Εισαγωγή γράφου από το xml αρχείο με όνομα `filename`.
- **public boolean** `exportToXML(String filename)`: Εξαγωγή του γράφου στο xml αρχείο με όνομα `filename`.
- **public** `HecataeusNodes getNodes()`: Εύρεση των κόμβων του γράφου.
- **public** `HecataeusEdges getEdges()`:Εύρεση των ακμών του γράφου.

- **public void** `resetKeyGenerator()`: Ανάθεση του μηδέν στην τιμή της μεταβλητής που δημιουργεί μοναδικά κλειδιά για τα στοιχεία του γράφου.
- **public int** `getKeyGenerator()`: Εύρεση της τιμής μεταβλητής που δημιουργεί μοναδικά κλειδιά για τα στοιχεία του γράφου.

Κλάσεις του πακέτου HecataeusEvolutionManager

4.2.8 Λειτουργίες της κλάσης EvolutionEventType

- **public static** `String convertEventTypeToString(int Value)`: Μετατροπή από την αναπαράσταση του τύπου γεγονότος με ακέραιο `Value` στην αναπαράσταση τύπου με συμβολοσειρά.
- **public static int** `convertStringToEventType(String Value)`: Μετατροπή από την αναπαράσταση του τύπου γεγονότος με συμβολοσειρά `Value` στην αναπαράσταση τύπου με ακέραιο.

4.2.9 Λειτουργίες της κλάσης EvolutionPolicyType

- **public static** `String convertPolicyTypeToString(int Value)` : Μετατροπή από την αναπαράσταση του τύπου πολιτικής με ακέραιο `Value` στην αναπαράσταση τύπου με συμβολοσειρά.
- **public static int** `convertStringToPolicyType(String Value)` : Μετατροπή από την αναπαράσταση του τύπου πολιτικής με συμβολοσειρά `Value` στην αναπαράσταση τύπου με ακέραιο.

4.2.10 Λειτουργίες της κλάσης EvolutionMessage

- `public EvolutionMessage(int SID, HecataeusEvolutionNode ns, HecataeusEvolutionNode nr, HecataeusEvolutionEvent event, int edgeType, int policyType)`: Δημιουργεί ένα καινούριο αντικείμενο μηνύματος με κόμβο αποστολέα , κόμβο παραλήπτη , γεγονός, τύπο ακμής μεταξύ κόμβου αποστολέα και κόμβου παραλήπτη και τύπος πολιτικής κόμβου αποστολέα .
- **public int** `get_SID()`: Επιστρέφει το μοναδικό αριθμό που χαρακτηρίζει τη συγκεκριμένη διαδικασία εύρεσης της επίδρασης ενός συγκεκριμένου γεγονότος
- **public** `HecataeusEvolutionNode get_ns()`: Επιστρέφει τον κόμβο αποστολέα.
- **public** `HecataeusEvolutionNode get_nr()`: Επιστρέφει τον κόμβο παραλήπτη.
- **public** `HecataeusEvolutionEvent get_event()`: Επιστρέφει το γεγονός.

- **public int** `get_edgeType()`: Επιστρέφει τον τύπο της ακμής μεταξύ του κόμβου αποστολέα και του κόμβου παραλήπτη.
- **public int** `get_policyType()`: Επιστρέφει τον τύπο πολιτικής του κόμβου αποστολέα.

4.2.11 Λειτουργίες της κλάσης EvolutionStatus

- **public static** `String convertStatusToString(int Value)` : Μετατροπή από την αναπαράσταση του τύπου κατάστασης με ακέραιο `Value` στην αναπαράσταση τύπου με συμβολοσειρά.
- **public static int** `convertStringToStatus(String Value)` : Μετατροπή από την αναπαράσταση του τύπου κατάστασης με συμβολοσειρά `Value` στην αναπαράσταση τύπου με ακέραιο.

4.2.12 Λειτουργίες της κλάσης HecataeusEvolutionNode

Εφόσον η κλάση κληρονομεί από την κλάση HecataeusNode, έχει τις ίδιες λειτουργίες με αυτήν, εκτός από τις παρακάτω, οι οποίες υπερσκελίζουν υπάρχουσες μεθόδους της κλάσης HecataeusNode:

- **public** `HecataeusEvolutionNode()`: Δημιουργία καινούργιου αντικειμένου κόμβου με δυνατότητα ορισμού πολιτικών.
- **public** `HecataeusEvolutionNode(String Name, int Type)`: Δημιουργία καινούργιου αντικειμένου κόμβου με δυνατότητα ορισμού πολιτικών με όνομα `Name` και τύπο `Type`.
- **public** `HecataeusEvolutionEdges getOutEdges()`: Εύρεση των ακμών που αναχωρούν από τον κόμβο.
- **public** `HecataeusEvolutionEdges getInEdges()`: Εύρεση των ακμών που εισέρχονται στον κόμβο.

Η κλάση HecataeusEvolutionNode προσφέρει επιπλέον τις παρακάτω λειτουργίες:

- **public void** `setPolicies(HecataeusEvolutionPolicies policies)`: Ανάθεση της συλλογής πολιτικών `policies` στον κόμβο.
- **public** `HecataeusEvolutionPolicies getPolicies()`: Εύρεση της συλλογής πολιτικών που έχουν οριστεί για τον κόμβο.
- **public void** `setEvents(HecataeusEvolutionEvents events)`: Ανάθεση της συλλογής γεγονότων `events` στον κόμβο.

- **public** `HecataeusEvolutionEvents getEvents()`: Εύρεση της συλλογής γεγονότων που έχουν οριστεί για τον κόμβο.
- **public void** `setStatus(int status)`: Ανάθεση του τύπου κατάστασης `status` στον κόμβο.
- **public int** `getStatus()`: Εύρεση του τύπου κατάστασης του κόμβου.

4.2.13 Λειτουργίες της κλάσης HecataeusEvolutionNodes

- **public** `HecataeusEvolutionNodes()`: Δημιουργία καινούργιου αντικειμένου λίστας κόμβων.
- **public** `HecataeusEvolutionNode getEvolutionNode(int Index)`: Εύρεση του κόμβου στη θέση `Index` στη λίστα κόμβων.
- **public** `HecataeusEvolutionNode setEvolutionNode (HecataeusEvolutionNode Value, int Index)` : Εισαγωγή του κόμβου `Value` στη θέση `Index` στη λίστα κόμβων.

4.2.14 Λειτουργίες της κλάσης HecataeuEvolutionGraph

Εφόσον η κλάση κληρονομεί από την κλάση HecataeusGraph, έχει τις ίδιες λειτουργίες με αυτήν, εκτός από τις παρακάτω, οι οποίες υπερσκελίζουν υπάρχουσες μεθόδους της κλάσης HecataeusGraph:

- **public** `HecataeusEvolutionGraph()`: Δημιουργία καινούριου αντικειμένου γράφου.
- **public boolean** `add_node(String Name, int Type)`: Εισαγωγή καινούριου κόμβου στο γράφο με όνομα `Name` και τύπο `Type`.
- **public boolean** `add_node(HecataeusEvolutionNode Node)`: Εισαγωγή του κόμβου `Node` στο γράφο.
- **public boolean** `remove_node(HecataeusEvolutionNode Node)`: Αφαίρεση του κόμβου `Node` από το γράφο.
- **public boolean** `add_edge(String Name, int Type, HecataeusEvolutionNode FromNode, HecataeusEvolutionNode ToNode)`: Εισαγωγή καινούριας ακμής στο γράφο με όνομα `Name`, τύπο `Type`, κόμβο πηγή `FromNode` και κόμβο προορισμού `ToNode`.
- **public boolean** `add_edge(HecataeusEvolutionEdge Edge)`: Εισαγωγή της ακμής `Edge` στο γράφο.

- **public boolean** `remove_edge(HecataeusEvolutionEdge Edge)`: Αφαίρεση της ακμής `Edge` από το γράφο.
- **public boolean** `clear_graph()` : Διαγραφή του γράφου και αφαίρεση όλων των στοιχείων του.
- **public** `HecataeusEvolutionNode getNode(String Key)`: Εύρεση του κόμβου του γράφου με τιμή κλειδιού `Key`.
- **public** `HecataeusEvolutionNode getNode(String Name,` **int** `Type)`: Εύρεση του κόμβου του γράφου με όνομα `Name` και τύπο `Type`.
- **public** `HecataeusEvolutionEdge getEdge(String Key)`: Εύρεση της ακμής του γράφου με τιμή κλειδιού `Key`.
- **public** `HecataeusEvolutionEdge getEdge(String Name,` **int** `Type)`: Εύρεση της ακμής του γράφου με όνομα `Name` και τύπο `Type`.
- **public** `HecataeusEvolutionNode createTableAlias(HecataeusEvolutionNode srcTable, String tblAlias)`: Δημιουργία ψευδωνύμου της σχέσης `srcTable` με όνομα `tblAlias`.
- **public** `HecataeusEvolutionNode getAttributeNode(String TableName, String AttributeName)`: Εύρεση ενός κόμβου τύπου «γνώρισμα» από το όνομα του `AttributeName` και το όνομα του κόμβου γονέα `TableName` (ο οποίος μπορεί να είναι τύπου «σχέση», «ερώτημα» ή «όψη»).
- **public boolean** `importFromXML(String filename)`: Εισαγωγή γράφου από το xml αρχείο με όνομα `filename`.
- **public boolean** `exportToXML(String filename)`: Εξαγωγή του γράφου στο xml αρχείο με όνομα `filename`.
- **public** `HecataeusEvolutionNodes getNodes()`: Εύρεση των κόμβων του γράφου.
- **public** `HecataeusEvolutionEdges getEdges()`: Εύρεση των ακμών του γράφου.

Η κλάση HecataeusEvolutionGraph προσφέρει επιπλέον τις παρακάτω λειτουργίες:

- **public** `HecataeusEvolutionNode getNode(String Name,` **int** `Type,` **int** `parentType)`: Εύρεση του κόμβου του γράφου με όνομα `Name`, τύπο `Type` και τύπο πατέρα `parentType`.
- **public void** `add_policy(HecataeusEvolutionNode node,` **int** `eventType, HecataeusEvolutionNode eventNode,` **int** `policyType)`: Προσθήκη μιας

πολιτική τύπου `policyType` που χειρίζεται ένα γεγονός τύπου `eventType`, το οποίο αφορά τον κόμβο `eventNode` στη συλλογή πολιτικών του κόμβου `node`.

- **`public void`** `add_policy(HecataeusEvolutionNode node, HecataeusEvolutionPolicy p)`: Προσθήκη της πολιτική `p` στη συλλογή πολιτικών του κόμβου `node`.
- **`public void`** `remove_policy(HecataeusEvolutionNode node, HecataeusEvolutionPolicy policy)`: Αφαίρεση της πολιτικής policy από τη συλλογή πολιτικών του κόμβου `node`.
- **`public void`** `add_event(HecataeusEvolutionNode eventNode, int eventType)`: Προσθήκη ενός γεγονότος τύπου `eventType` στη συλλογή γεγονότων του κόμβου `eventNode`.
- **`public boolean`** `add_event(HecataeusEvolutionNode eventNode, HecataeusEvolutionEvent e)`: Προσθήκη του γεγονότος `e` στη συλλογή γεγονότων του κόμβου `eventNode`.
- **`public void`** `remove_event(HecataeusEvolutionNode eventNode, HecataeusEvolutionEvent event)`: Αφαίρεση του γεγονότος `event` από τη συλλογή γεγονότων του κόμβου `eventNode`.
- **`public void`** `getInNodes(HecataeusEvolutionNode node, HecataeusEvolutionNodes InNodes)`: Εύρεση όλων των κόμβων που βρίσκονται σε όλα τα μονοπάτια που καταλήγουν στον κόμβο `node` και εισαγωγή τους στη συλλογή κόμβων `InNodes`.
- **`private void`** `getInEdges(HecataeusEvolutionNode node, HecataeusEvolutionEdges InEdges)`: Εύρεση όλων των ακμών που βρίσκονται σε όλα τα μονοπάτια που καταλήγουν στον κόμβο `node` και εισαγωγή τους στη συλλογή ακμών `InEdges`.
- **`public`** `HecataeusEvolutionNodes getAffectedNodes(HecataeusEvolutionNode affecting_node, int eventType)`: Εύρεση όλων των κόμβων που επηρεάζονται από ένα γεγονός τύπου `eventType` που αφορά τον κόμβο `affecting_node`.
- **`public`** `HecataeusEvolutionEdges getAffectedEdges(HecataeusEvolutionNode affecting_node, int eventType)`: Εύρεση όλων των ακμών που επηρεάζονται από ένα γεγονός τύπου eventType που αφορά τον κόμβο `affecting_node`.

- **public void** `initializeChange(HecataeusEvolutionEvent event)`: Συνάρτηση που κάνει τις απαραίτητες αρχικοποιήσεις, πριν την κλήση της συνάρτησης `propagateChanges()`, με γεγονός `event`. Είναι η συνάρτηση που καθορίζει η προεπιλεγμένη πολιτική να είναι προώθηση της αλλαγής.
- **public void** `propagateChanges(Queue queue, EvolutionMessage currentMessage)`: Συνάρτηση που βρίσκει τα στοιχεία του γράφου που επηρεάζονται από ένα γεγονός και θέτει την κατάσταση τους, ανάλογα με το πώς επηρεάζονται από το γεγονός. Καλείται από την `initializeChange()`.
- **public int** `determinePolicy(HecataeusEvolutionEvent event, HecataeusEvolutionNode ns, HecataeusEvolutionNode nr, int previousPolicyType)`: Συνάρτηση που επιστρέφει την πολιτική που θα καθορίσει την κατάσταση του τρέχοντος κόμβου. Καλείται από την `propagateChanges()`.
- **public int** `getPrevailingPolicy(HecataeusEvolutionNode ns,HecataeusEvolutionNode nr,int previousPolicyType, int nrPolicyType)`: Εύρεση της επικρατούσας πολιτικής σε περίπτωση σύγκρουσης πολιτικών. Καλείται από την `determinePolicy()`.
- **public int** `getParentNodeType(HecataeusEvolutionNode node)`: Εύρεση του τύπου του κόμβου-πατέρα του κόμβου `node`. Η συνάρτηση χρησιμοποιείται στην περίπτωση που ο κόμβος `node` είναι τύπου γνωρίσματος, για να επιστρέψει αν είναι γνώρισμα σχέσης, όψης ή ερωτήματος.
- **public int** `getParentNode (HecataeusEvolutionNode node)`: Εύρεση του κόμβου-πατέρα του κόμβου `node`. Η συνάρτηση χρησιμοποιείται στην περίπτωση που ο κόμβος `node` είναι τύπου γνωρίσματος, για να επιστρέψει τη σχέση, την όψη ή το ερώτημα.

4.2.15 Λειτουργίες της κλάσης HecataeusEvolutionEdge

Εφόσον η κλάση κληρονομεί από την κλάση HecataeusEdge, έχει τις ίδιες λειτουργίες με αυτήν, εκτός από τις παρακάτω, οι οποίες υπερσκελίζουν υπάρχουσες μεθόδους της κλάσης HecataeusEdge:

- **public** `HecataeusEvolutionEdge()`: Δημιουργία καινούργιου αντικειμένου ακμής.
- **public** `HecataeusEvolutionEdge(String name, int type, HecataeusEvolutionNode fromNode, HecataeusEvolutionNode toNode)`: Δημιουργία καινούργιου αντικειμένου ακμής με όνομα Name, τύπο Type, κόμβο πηγή `FromNode` και κόμβο προορισμού `ToNode`.

- **public void** `setFromNode(HecataeusEvolutionNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος πηγή» της ακμής.
- **public void** `setToNode(HecataeusEvolutionNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος προορισμού» της ακμής.
- **public** `HecataeusEvolutionNode getFromNode()`:Εύρεση της τιμής της ιδιότητας «κόμβος πηγή» της ακμής.
- **public** `HecataeusEvolutionNode getToNode()`: Εύρεση της τιμής της ιδιότητας «κόμβος προορισμού» της ακμής.

Η κλάση HecataeusEvolutionEdge προσφέρει επιπλέον τις παρακάτω λειτουργίες:

- **public void** `setStatus(int status)`: Ανάθεση του τύπου κατάστασης `status` στην ακμή.
- **public int** `getStatus()`: Εύρεση του τύπου κατάστασης της ακμής.

4.2.16 Λειτουργίες της κλάσης HecataeusEvolutionEdges

- **public** `HecataeusEvolutionEdges()`: Δημιουργία καινούργιου αντικειμένου λίστας ακμών.
- **public** `HecataeusEvolutionEdge getEvolutionEdge(int Index)`: Εύρεση της ακμής στη θέση `Index` στη λίστα ακμών.
- **public** `HecataeusEvolutionEdge setEvolutionEdge(HecataeusEvolutionEdge Value, int Index)`: Εισαγωγή της ακμής `Value` στη θέση `Index` στη λίστα ακμών.

4.2.17 Λειτουργίες της κλάσης HecataeusEvolutionPolicy

- **public** `HecataeusEvolutionPolicy()`: Δημιουργία καινούριου αντικειμένου πολιτικής.
- **public** `HecataeusEvolutionPolicy(int eventType, HecataeusEvolutionNode eventNode, int policyType)`: Δημιουργία καινούριου αντικειμένου πολιτικής τύπου `policyType`, για γεγονός τύπου `eventType` στον κόμβο `eventNode`.
- **public int** `getEventType()`: Εύρεση του τύπου του γεγονότος το οποίο χειρίζεται η πολιτική.
- **public void** `setEventType(int Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «τύπος γεγονότος» της πολιτικής.

- **public** `HecataeusEvolutionNode getEventNode()`: Εύρεση του κόμβου τον οποίο αφορά το γεγονός που χειρίζεται η πολιτική.
- **public void** `setEventNode(HecataeusEvolutionNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος γεγονότος» της πολιτικής.
- **public int** `getPolicyType()`:Εύρεση του τύπου της πολιτικής.
- **public void** `setPolicyType(int Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «τύπος πολιτικής» της πολιτικής.

4.2.18 Λειτουργίες της κλάσης HecataeusEvolutionPolicies

- **public** `HecataeusEvolutionPolicies()`: Δημιουργία καινούριου αντικειμένου λίστας πολιτικών.
- **public** `HecataeusEvolutionPolicy getEvolutionPolicy(int Index)`: Εύρεση της πολιτικής στη θέση `Index` στη λίστα πολιτικών.
- **public** `HecataeusEvolutionPolicy getPolicy(int eventType, int policyType)`: Εύρεση της πολιτικής με τύπο `policyType` και τύπο γεγονότος `eventType` από τη λίστα πολιτικών.
- **public** `HecataeusEvolutionPolicy setEvolutionPolicy(HecataeusEvolutionPolicy Value, int Index)`: Εισαγωγή της πολιτικής `Value` στη θέση `Index` στη λίστα πολιτικών.

4.2.19 Λειτουργίες της κλάσης HecataeusEvolutionEvent

- **public** `HecataeusEvolutionEvent(HecataeusEvolutionNode node,int type)`: Δημιουργία καινούριου γεγονότος τύπου στον κόμβο .
- **public void** `setEventType(int type)`: Ανάθεση του τύπου γεγονότος `type` στην ιδιότητα «τύπος γεγονότος» του γεγονότος.
- **public int** `getEventType()`:Εύρεση του τύπου γεγονότος.
- **public void** `setEventNode(HecataeusEvolutionNode node)`: Ανάθεση του κόμβου γεγονότος `node` στην ιδιότητα «κόμβος γεγονότος» του γεγονότος.
- **public** `HecataeusEvolutionNode getEventNode()`:Εύρεση του κόμβου που αφορά (και στον οποίο ορίζεται) το γεγονός.

4.2.20 Λειτουργίες της κλάσης HecataeusEvolutionEvents

- **public** `HecataeusEvolutionEvents()`: Δημιουργία καινούριας συλλογής γεγονότων.
- **ublic** `HecataeusEvolutionEvent getEvent(`**int** `Index)`: Εύρεση του γεγονότος στη θέση `Index` στη λίστα γεγονότων.
- **public** `HecataeusEvolutionEvent getEventByType(`**int** `type)`: Εύρεση του γεγονότος με τύπο `type`.
-

Κλάσεις του πακέτου HecataeusJungGraph

4.2.21 Λειτουργίες της κλάσης HecataeusJungEdge

- **public** `HecataeusJungEdge(Vertex from, Vertex to)`: Δημιουργία καινούργιου αντικειμένου ακμής με δυνατότητα οπτικοποίησης.
- **public** `HecataeusJungEdge(String name,` **int** `type , HecataeusJungNode fromNode, HecataeusJungNode toNode)`: Δημιουργία καινούργιου αντικειμένου ακμής με όνομα Name, τύπο Type, κόμβο πηγή `FromNode` και κόμβο προορισμού `ToNode`.
- **public** `HecataeusEvolutionEdge getHecataeusEvolutionEdge()`: Εύρεση του αντίστοιχου αντικειμένου HecataeusEvolutionEdge.
- **public void** `setIsAffected(Boolean isAffected)`: Ανάθεση της τιμής `isAffected` στην ιδιότητα «επηρεασμένο» της ακμής. Χρειάζεται για το χρωματικό τονισμό των ακμών του γράφου που επηρεάζονται από ένα γεγονός (κίτρινες).
- **public** `String getName()`:Εύρεση της τιμής της ιδιότητας «όνομα» της ακμής.
- **public void** `setName(String Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «όνομα» της ακμής.
- **public int** `getType()`:Εύρεση της τιμής της ιδιότητας «τύπος» της ακμής.
- **public void** `setType(`**int** `Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «τύπος» της ακμής.
- **public** `String getKey()`:Εύρεση της τιμής της ιδιότητας «κλειδί» της ακμής.
- **public void** `setKey(String Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κλειδί» της ακμής.

- **public** `HecataeusJungNode getFromNode()`: Εύρεση της τιμής της ιδιότητας «κόμβος πηγή» της ακμής.
- **public void** `setFromNode(HecataeusJungNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος πηγή» της ακμής.
- **public** `HecataeusJungNode getToNode()`: Εύρεση της τιμής της ιδιότητας «κόμβος προορισμού» της ακμής.
- **public void** `setToNode(HecataeusJungNode Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «κόμβος προορισμού» της ακμής.

4.2.22 Λειτουργίες της κλάσης HecataeusJungEdges

- **public** `HecataeusJungEdges()`: Δημιουργία καινούργιου αντικειμένου λίστας ακμών.
- **public** `HecataeusJungEdge getJungEdge(int Index)`: Εύρεση της ακμής στη θέση `Index` στη λίστα ακμών.
- **public** `HecataeusJungEdge setJungEdge(HecataeusJungEdge Value, int Index)`: Εισαγωγή της ακμής `Value` στη θέση `Index` στη λίστα ακμών.

4.2.23 Λειτουργίες της κλάσης HecataeusJungNode

- **public** `HecataeusJungNode()`: Δημιουργία ενός καινούργιου αντικειμένου κόμβου με δυνατότητα οπτικοποίησης.
- **public** `HecataeusJungNode(String name, int type)`: Δημιουργία καινούργιου αντικειμένου κόμβου με όνομα `Name` και τύπο `Type`.
- **public** `HecataeusEvolutionNode getHecataeusEvolutionNode()`: Εύρεση του αντικειμένου HecataeusEvolutionNode το οποίο αντιστοιχεί σ' αυτό το αντικείμενο HecataeusJungNode.
- **public void** `setLocation(int x,int y)`: Ανάθεση των συντεταγμένων `x` και `y` στην ιδιότητα «θέση» του κόμβου.
- **public** `Point2D getLocation()`: Εύρεση της τιμής της ιδιότητας «θέση» του κόμβου.
- **public void** `setLastChildLocation(int x,int y)`: Ανάθεση των συντεταγμένων `x` και `y` στην ιδιότητα «θέση του τελευταίου κόμβου-παιδιού» του κόμβου. Χρησιμοποιείται στην περίπτωση που ο κόμβος είναι τύπου «σχέση», «ερώτημα»

ή «όψη» για να κρατά τη θέση του τελευταίου κόμβου «γνώρισμα» που προστίθεται για να οπτικοποιηθεί ο γράφος στο πλάνο.

- **public** `Point2D getLastChildLocation()`: Εύρεση της τιμής της ιδιότητας «θέση του τελευταίου κόμβου-παιδιού» του κόμβου.
- **public void** `setHasPolicies(Boolean hasPolicies)`: Ανάθεση της τιμής `hasPolicies` στην ιδιότητα «έχει πολιτικές» του κόμβου. Χρειάζεται για τον διαφορετικό χειρισμό των κόμβων του γράφου για τους οποίους έχουν οριστεί πολιτικές.
- **public** `Boolean getHasPolicies()`:Εύρεση της τιμής της ιδιότητας «έχει πολιτικές» του κόμβου.
- **public void** `setHasEvents(Boolean hasEvents)`: Ανάθεση της τιμής `hasEvents` στην ιδιότητα «έχει γεγονότα» του κόμβου. Χρειάζεται για τον διαφορετικό χειρισμό των κόμβων του γράφου για τους οποίους έχουν οριστεί γεγονότα.
- **public** `Boolean getHasEvents()`: Εύρεση της τιμής της ιδιότητας «έχει γεγονότα» του κόμβου.
- **public** `String getName()`: Εύρεση της τιμής της ιδιότητας «όνομα» του κόμβου.
- **public void** `setName(String Value)`: Ανάθεση της τιμής `Value` στην ιδιότητα «όνομα» του κόμβου.
- **public int** `getType()` : Εύρεση της τιμής της ιδιότητας «τύπος» του κόμβου.
- **public void** `setType(int Value)` : Ανάθεση της τιμής `Value` στην ιδιότητα «τύπος» του κόμβου.
- **public** `String getKey()` : Εύρεση της τιμής της ιδιότητας «κλειδί» του κόμβου.
- **public void** `setKey(String Value)` : Ανάθεση της τιμής `Value` στην ιδιότητα «κλειδί» του κόμβου.

4.2.24 Λειτουργίες της κλάσης HecataeusJungNodes

- **public** `HecataeusJungNodes()`: Δημιουργία καινούργιου αντικειμένου λίστας κόμβων.
- **public** `HecataeusJungNode getJungNode(int Index)`: Εύρεση του κόμβου στη θέση `Index` στη λίστα κόμβων.
- **public** `HecataeusJungNode setJungNode(HecataeusJungNode Value, int Index)`: Εισαγωγή του κόμβου `Value` στη θέση `Index` στη λίστα κόμβων.
-

4.2.25 Λειτουργίες της κλάσης HecataeusJungGraph

- **public** `HecataeusJungGraph()`: Δημιουργία καινούριου αντικειμένου γράφου.
- **public** `HecataeusEvolutionGraph getHecataeusEvolutionGraph()`:Εύρεση του αντικειμένου HecataeusEvolutionGraph το οποίο αντιστοιχεί σ' αυτό το αντικείμενο HecataeusJungGraph.
- **public** `HecataeusJungNode getJungNodeFromEvolution(HecataeusEvolutionNode hecEvNode)`: Εύρεση του αντικειμένου HecataeusJungNode το οποίο αντιστοιχεί στο αντικείμενο `hecEvNode`.
- **public** `HecataeusJungEdge getJungEdgeFromEvolution(HecataeusEvolutionEdge hecEvEdge)`: Εύρεση του αντικειμένου HecataeusJungEdge το οποίο αντιστοιχεί στο αντικείμενο `hecEvEdge`.
- **public** `HecataeusEvolutionNodes getEvolutionNodesFromJung(HecataeusJungNodes hecJuNodes)`: Εύρεση της λίστας HecataeusEvolutionNodes, της οποίας τα στοιχεία αντιστοιχούν στα στοιχεία της λίστας `hecJuNodes`.
- **public** `HecataeusEvolutionEdges getEvolutionEdgesFromJung(HecataeusJungEdges hecJuEdges)`: Εύρεση της λίστας HecataeusEvolutionEdges, της οποίας τα στοιχεία αντιστοιχούν στα στοιχεία της λίστας `hecJuEdges`.
- **public** `HecataeusJungNodes getJungNodesFromEvolution(HecataeusEvolutionNodes hecEvNodes)`: Εύρεση της λίστας HecataeusJungNodes, της οποίας τα στοιχεία αντιστοιχούν στα στοιχεία της λίστας `hecEvNodes`.
- **public** `HecataeusJungEdges getJungEdgesFromEvolution(HecataeusEvolutionEdges hecEvEdges)`: Εύρεση της λίστας HecataeusJungEdges, της οποίας τα στοιχεία αντιστοιχούν στα στοιχεία της λίστας `hecEvEdges`.
- **public boolean** `add_node(String Name, int Type)`: Εισαγωγή καινούριου κόμβου στο γράφο με όνομα `Name` και τύπο `Type`.
- **public boolean** `add_node(HecataeusJungNode node)`: Εισαγωγή του κόμβου `node` στο γράφο.
- **public boolean** `remove_node(HecataeusJungNode node)`: Αφαίρεση του κόμβου `node` από το γράφο.

- **public boolean** `add_edge(String Name, int Type, HecataeusJungNode FromNode, HecataeusJungNode ToNode)`: Εισαγωγή καινούριας ακμής στο γράφο με όνομα `Name`, τύπο `Type`, κόμβο πηγή `FromNode` και κόμβο προορισμού `ToNode`.
- **public boolean** `add_edge(HecataeusJungEdge edge)`: Εισαγωγή της ακμής `edge` στο γράφο.
- **public boolean** `remove_edge(HecataeusJungEdge edge)`: Αφαίρεση της ακμής `edge` από το γράφο.
- **public boolean** `clear_graph()` : Διαγραφή του γράφου.
- **public** `HecataeusJungNode getNode(String Key)`: Εύρεση του κόμβου του γράφου με τιμή κλειδιού `Key`.
- **public** `HecataeusJungNode getNode(String Name, int Type)`: Εύρεση του κόμβου του γράφου με όνομα `Name` και τύπο `Name`.
- **public** `HecataeusJungEdge getEdge(String Key)`: Εύρεση της ακμής του γράφου με τιμή κλειδιού `Key`.
- **public** `HecataeusJungEdge getEdge(String Name, int Type)`: Εύρεση της ακμής του γράφου με όνομα `Name` και τύπο `Name`.
- **public** `ArrayList<HecataeusJungNode> importNodesFromXML(File file)`: Εισαγωγή των κόμβων του γράφου από το xml αρχείο `file`.
- **public** `ArrayList<HecataeusJungEdge> importEdgesFromXML(File file)`: Εισαγωγή των ακμών του γράφου από το xml αρχείο `file`. Αυτή είναι η περίπτωση, στην οποία, αν υπάρχει ήδη γράφος, αυτός σβήνεται και στη συνέχεια εισάγεται ο νέος γράφος από το xml αρχείο.
- **public** `ArrayList<HecataeusJungEdge> addSubgraphEdgesFromXML(File file,int maxKey)`: Εισαγωγή των ακμών του γράφου από το xml αρχείο `file`. Αυτή είναι η περίπτωση, στην οποία, αν υπάρχει ήδη γράφος, αυτός παραμένει, ενώ εισάγεται και ο νέος γράφος από το xml αρχείο.
- **public boolean** `exportToXML(File file)`: Εξαγωγή του γράφου στο xml αρχείο με όνομα `filename`.
- **public** `HecataeusJungNodes getJungNodes())`: Εύρεση των κόμβων του γράφου.
- **public** `HecataeusJungEdges getJungEdges()`:Εύρεση των ακμών του γράφου.

Κλάσεις του πακέτου Hecataeus

4.2.26 Λειτουργίες της κλάσης HecataeusModalGraphMouse

- **public** `HecataeusModalGraphMouse()`: Δημιουργία καινούριου αντικειμένου ποντικιού με δύο καταστάσεις λειτουργίας: μετακίνηση και επιλογή.
- **public** `HecataeusModalGraphMouse(float in, float out)`: Δημιουργία καινούριου αντικειμένου ποντικιού με δύο καταστάσεις λειτουργίας (μετακίνηση και επιλογή) και τιμές για μεγέθυνση και σμίκρυνση `in` και `out`, αντίστοιχα.
- **public void** `setMode(Mode mode)`: Αλλαγή της κατάστασης σε `mode`.
- **public** `JComboBox getModeComboBox()`:Επιστρέφει μια λίστα επιλογών με τις δύο πιθανές τιμές για επιλογή της τρέχουσας κατάστασης του ποντικιού.
- **public** `JMenu getModeMenu()`: Επιστρέφει ένα μενού επιλογής για επιλογή της τρέχουσας κατάστασης του ποντικιού.

4.2.27 Λειτουργίες της κλάσης JungEditingPopupGraphMousePlugin

- **public** `JungEditingPopupGraphMousePlugin(SettableVertexLocationFunction vertexLocations)`: Δημιουργία καινούργιου αντικειμένου για επιπρόσθετη λειτουργικότητα του ποντικιού, που λαμβάνει υπόψη τη θέση των κόμβων του γράφου.
- **public void** `mouseClicked(MouseEvent e)`: Λειτουργία όταν πατηθεί το ποντίκι σε κενή περιοχή (όχι πάνω σε κόμβο ή ακμή). Κάθε στοιχείο του γράφου που ήταν επιλεγμένο χάνει αυτή την ιδιότητα . Επίσης σε περίπτωση που άλλαξε η κατάσταση των κόμβων λόγω ενός γεγονότος, επανέρχεται η κατάσταση κάθε κόμβου στην αρχική.
- **protected void** `handlePopup(MouseEvent e)`: Λειτουργία όταν πατηθεί το δεξί πλήκτρο του ποντικιού. Οι διαθέσιμες επιλογές εμφανίζονται σε ένα εμφανιζόμενο μενού επιλογής και είναι αυτές που παρουσιάστηκαν στην παράγραφο 4.1.25.
- **public void** `setFlagForPoliciesExist(boolean value)`: Ανάθεση της τιμή `value` στη λογική μεταβλητή που δηλώνει αν υπάρχουν πολιτικές.
- **public boolean** `getFlagForPoliciesExist()`: Εύρεση της τιμής της λογικής μεταβλητής που δηλώνει αν υπάρχουν πολιτικές.
- **public void** `setFlagForEventsExist(boolean value)`: Ανάθεση της τιμή `value` στη λογική μεταβλητή που δηλώνει αν υπάρχουν γεγονότα.
- **public boolean** `getFlagForEventsExist()`:Εύρεση της τιμής της λογικής μεταβλητής που δηλώνει αν υπάρχουν γεγονότα.

4.2.28 Λειτουργίες της κλάσης HecataeusJungViewer

- **public** `HecataeusJungViewer()`: Δημιουργία καινούριου αντικειμένου του βασικού παραθύρου της εφαρμογής.
- **public void** `importFromXML(File file)`: Εισάγει τις πληροφορίες για ένα γράφο από το xml αρχείο `file` και τον εμφανίζει στο πλάνο, σβήνοντας ότι προϋπάρχει στο πλάνο.

public void `importFromScript(HecataeusEvolutionGraph evGraph, HecataeusJungGraph graph)`: Εισάγει τις πληροφορίες για ένα γράφο απευθείας από αρχεία τύπου sql, σβήνοντας ό,τι υπάρχει στο πλάνο. Συγκεκριμένα, εισάγεται το αρχείο που περιέχει τον ορισμό των δεδομένων (DDL) και το αρχείο που περιέχει τις όψεις και τα ερωτήματα (SQL).

- **public void** `addGraphFromXML(File file)`: Εισάγει τις πληροφορίες για ένα γράφο από το xml αρχείο `file` και τον εμφανίζει στο πλάνο, κρατώντας ότι προϋπάρχει στο πλάνο.
- **public void** `addEvent()`: Εμφανίζει ένα παράθυρο για ορισμό νέου γεγονότος, παίρνει τις πληροφορίες από το παράθυρο και ορίζει το επιλεγμένο γεγονός στον επιλεγμένο κόμβο.
- **public void** `setRenderer(Renderer pr)`: Θέτει τον `pr` ως τον Renderer (αντικείμενο που εμφανίζει το γράφο στο πλάνο) της εφαρμογής.
- **public** `PluggableRenderer getRenderer()`: Επιστρέφει το αντικείμενο Renderer που εμφανίζει το γράφο στο πλάνο.
- **public void** `writeJPEGImage(File file)`: Αποθήκευση του πλάνου ως εικόνα στο `file`.

4.3 Κωδικοποίηση αρχείων

Το σύστημα μπορεί να εισάγει ένα γράφο ή να εξάγει ένα γράφο σε αρχείο xml. Ο γράφος κωδικοποιείται σε αρχείο xml ως εξής:

Ο γράφος περιέχεται μεταξύ των στοιχείων σήμανσης <HGraph> και </HGraph>.

Οι κόμβοι του γράφου περιέχονται μεταξύ των στοιχείων σήμανσης <HNodes> και </HNodes>, ενώ οι ακμές μεταξύ των στοιχείων σήμανσης <HEdges> και </HEdges>, τα οποία εντίθενται μέσα στα στοιχεία σήμανσης του γράφου.

Κάθε κόμβος περικλείεται μέσα στα στοιχεία σήμανσης <HNode> και </HNode> που εντίθενται μέσα στα <HNodes> και </HNodes>, ενώ οι ακμές μεταξύ των στοιχείων σήμανσης <HEdge> και </HEdge> που εντίθενται μέσα στα <HEdges> και </HEdges>.

Κάθε κόμβος έχει ως ιδιότητες το κλειδί του, το όνομά του και τον τύπο του, τα οποία περικλείονται στα στοιχεία σήμανσης <HKey> και </HKey>, <HName> και </HName> και <HType> και </HType >, αντίστοιχα. Αυτά τα σημεία σήμανσης εντίθενται μέσα στα <HNode> και </HNode> του συγκεκριμένου κόμβου. Το κλειδί είναι ένας θετικός ακέραιος, το όνομα είναι μια συμβολοσειρά (χωρίς εισαγωγικά) και ο τύπος μια συμβολοσειρά που αναπαριστά τύπο κόμβου, όπως περιγράφηκε στην παράγραφο 4.1.4.

Κάθε ακμή έχει τις ίδιες ιδιότητες με τον κόμβο που περικλείονται μεταξύ των ίδιων στοιχείων σήμανσης. Επιπλέον όμως σ' αυτές τις ιδιότητες, μια ακμή έχει ως ιδιότητες και τον κόμβο από τον οποίο αναχωρεί και τον κόμβο στον οποίο καταλήγει. Οι ιδιότητες αυτές είναι θετικοί ακέραιοι που αναπαριστούν τα κλειδιά των αντίστοιχων κόμβων και περιλαμβάνονται μεταξύ των σημάνσεων <FromNode> και </FromNode> και <ToNode> και </ToNode>.

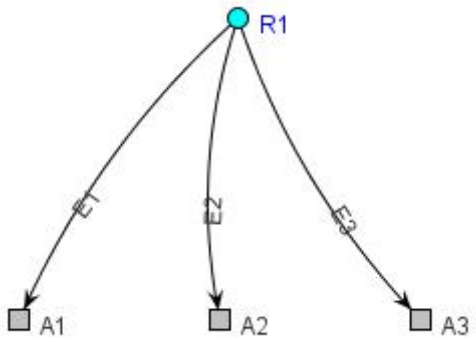
Μέσα στα στοιχεία σήμανσης <HGraph> και </HGraph> εντίθενται επίσης τα στοιχεία σήμανσης <HKeyGen> και </ HKeyGen> που περιλαμβάνουν την τιμή (θετικός ακέραιος) του πλήθους των στοιχείων του γράφου.

Ένα απλό παράδειγμα φαίνεται στο παρακάτω αρχείο xml, το οποίο περιγράφει το γράφο της Εικόνας 4.7.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <HGraph>
  - <HNodes>
    - <HNode>
        <Key>1</Key>
        <Name>R1</Name>
        <Type>NODE_TYPE_RELATION</Type>
      </HNode>
    - <HNode>
        <Key>2</Key>
        <Name>A1</Name>
        <Type>NODE_TYPE_ATTRIBUTE</Type>
      </HNode>
    - <HNode>
        <Key>3</Key>
        <Name>A2</Name>
```

```
        <Type>NODE_TYPE_ATTRIBUTE</Type>
      </HNode>
    - <HNode>
        <Key>4</Key>
        <Name>A3</Name>
        <Type>NODE_TYPE_ATTRIBUTE</Type>
      </HNode>
    </HNodes>
  - <HEdges>
    - <HEdge>
        <Key>5</Key>
        <Name>E1</Name>
        <Type>EDGE_TYPE_SCHEMA</Type>
        <FromNode>1</FromNode>
        <ToNode>2</ToNode>
      </HEdge>
    - <HEdge>
        <Key>6</Key>
        <Name>E2</Name>
        <Type>EDGE_TYPE_SCHEMA</Type>
        <FromNode>1</FromNode>
        <ToNode>3</ToNode>
      </HEdge>
    - <HEdge>
        <Key>7</Key>
        <Name>E3</Name>
        <Type>EDGE_TYPE_SCHEMA</Type>
        <FromNode>1</FromNode>
        <ToNode>4</ToNode>
      </HEdge>
    </HEdges>
    <HKeyGen>7</HKeyGen>
  </HGraph>
```

Εικόνα 4.6. Αρχείο xml που αναπαριστά το γράφο της Εικόνας 4.7.



Εικόνα 4.7. Ο γράφος που αναπαριστάται στο xml αρχείο της Εικόνας 4.6.

Εάν ένας κόμβος έχει ορισμένες πολιτικές, τότε μέσα στα στοιχεία σήμανσης <HNode> και </HNode> εντίθενται τα στοιχεία σήμανσης <HPolicies> και </HPolicies>. Κάθε πολιτική περιέχεται εντός των στοιχείων σήμανσης <HPolicy> και </HPolicy>, που εντίθενται μέσα στα στοιχεία σήμανσης <HPolicies> και </HPolicies>. Μια πολιτική έχει ως ιδιότητες τον κόμβο τον οποίο αφορά το γεγονός που χειρίζεται η πολιτική, τον τύπο του γεγονότος και τον τύπο της πολιτικής, οι οποίες περικλείονται στα στοιχεία σήμανσης <EventNode> και </EventNode>, <EventType> και </EventType> και <PolicyType> και </PolicyType>, αντίστοιχα. Ο κόμβος που αφορά το γεγονός που χειρίζεται η πολιτική είναι ένας θετικός ακέραιος, που αναπαριστά το κλειδί του κόμβου, ο τύπος του γεγονότος είναι μια συμβολοσειρά που αναπαριστά τον τύπο του γεγονότος, όπως περιγράφεται στην παράγραφο 4.1.8 και ο τύπος της πολιτικής είναι μια συμβολοσειρά τον τύπο της πολιτικής, όπως περιγράφεται στην παράγραφο 4.1.9.

Ομοίως, αν στον κόμβο έχουν οριστεί γεγονότα, κάθε γεγονός περιέχεται εντός των στοιχείων σήμανσης <HEvent> και </HEvent>, που εντίθενται μέσα στα στοιχεία σήμανσης <HEvents> και </HEvents>. Ένα γεγονός έχει ως ιδιότητες τον κόμβο τον οποίο αφορά το γεγονός που χειρίζεται η πολιτική και τον τύπο του, οι οποίες περικλείονται στα στοιχεία σήμανσης <EventNode> και </EventNode> και <EventType> και </EventType>, αντίστοιχα. Ο κόμβος που αφορά το γεγονός που χειρίζεται η πολιτική είναι ένας θετικός ακέραιος, που αναπαριστά το κλειδί του κόμβου και ο τύπος του γεγονότος είναι μια συμβολοσειρά που αναπαριστά τον τύπο του γεγονότος, όπως περιγράφεται στην παράγραφο 4.1.8.

Έστω ο κόμβος που αναπαριστά μια σχέση, με κλειδί 1, όνομα R1, ο οποίος έχει μια πολιτική «Σε περίπτωση διαγραφής ενός γνωρίσματος στον κόμβο με κλειδί 2, τότε εμπόδισε την αλλαγή». Ο κόμβος αυτός αναπαριστάται σε xml ως εξής:

```
- <HNode>
  <Key>1</Key>
  <Name>R1</Name>
```

```
    <Type>NODE_TYPE_RELATION</Type>
  - <HPolicies>
  - <HPolicy>
    <eventNode>2</eventNode>
    <eventType>Remove_Attribute</eventType>
    <policyType>Block</policyType>
    </HPolicy>
    </HPolicies>
    </HNode>
```

Εικόνα 4.8. Αναπαράσταση κόμβου με πολιτική σε αρχείο xml.

Έστω ο κόμβος που αναπαριστά μια σχέση, με κλειδί 2, όνομα Α1, ο οποίος έχει ορισμένο ένα γεγονός «Διαγραφής γνωρίσματος» που αφορά τον ίδιο κόμβο. Ο κόμβος αυτός αναπαριστάται σε xml ως εξής:

```
- <HNode>
    <Key>2</Key>
    <Name>A1</Name>
    <Type>NODE_TYPE_ATTRIBUTE</Type>
  - <HEvents>
    - <HEvent>
        <eventNode>2</eventNode>
        <eventType>Remove_Attribute</eventType>
      </HEvent>
    </HEvents>
  </HNode>
- <HNode>
```

Εικόνα 4.9. Αναπαράσταση κόμβου με πολιτική σε αρχείο xml.

5 Υλοποίηση

Στην ενότητα αυτή θα παρουσιαστούν θέματα υλοποίησης του συστήματος. Συγκεκριμένα, στην παράγραφο 5.1 παρουσιάζονται η γλώσσα προγραμματισμού στην οποία γράφτηκε η εφαρμογή, το περιβάλλον ανάπτυξης της εφαρμογής και η βιβλιοθήκη για την αναπαράσταση του γράφου που χρησιμοποιήθηκε, καθώς και οι λόγοι επιλογής τους. Στην παράγραφο 5.2 περιγράφονται οι απαιτήσεις για την εγκατάσταση του συστήματος ΕΚΑΤΑΙΟΣ ΙΙ σε υπολογιστή.

5.1 Πλατφόρμες και προγραμματιστικά εργαλεία

Η εφαρμογή γράφτηκε στη γλώσσα προγραμματισμού Java. Ακολουθούν λίγα λόγια για την Java και οι λόγοι επιλογής της.

5.1.1 Η γλώσσα προγραμματισμού: JAVA

Η εφαρμογή γράφτηκε στη γλώσσα προγραμματισμού Java. Η Java είναι μια αντικειμενοστραφής γλώσσα που βασίστηκε στην C++. Είναι μια γλώσσα ανοικτού κώδικα και ανεξάρτητη πλατφόρμας, γεγονότα συνέβαλλαν στην ευρεία διάδοσή της και εμπλουτισμού της με βιβλιοθήκες. Η Java δεν παρέχει πρόσβαση χαμηλού επιπέδου όπως η C++. Στους στόχους της γλώσσας ήταν να κρατήσει τα καλά χαρακτηριστικά της C++, προσφέροντας όμως μεγαλύτερη ευκολία μάθησης και μικρότερη πιθανότητα σφαλμάτων, καθώς:

- Φροντίζει αυτόματα για τη δέσμευση και αποδέσμευση μνήμης.
- Δεν περιλαμβάνει δείκτες.
- Περιλαμβάνει μόνο μονή κληρονομικότητα.

Επίσης, η Java θεωρείται ασφαλής γλώσσα, καθώς όταν ένα πρόγραμμα εκτελείται από ένα διερμηνευτή Java ενός προγράμματος περιήγησης, δεν υπάρχει δυνατότητα χρήσης χαρακτηριστικών της γλώσσας που θα μπορούσαν να χρησιμοποιηθούν για κακόβουλη χρήση.

Η Java αναπτύσσεται διαρκώς, προσφέροντας επιπλέον δυνατότητες στους προγραμματιστές της [CadL03], [Wiki07].

5.1.1.1 Λόγοι επιλογής της Java

Η εφαρμογή αποτελείται από διακριτές έννοιες που αναπαρίστανται εύκολα ως αντικείμενα, όπως «γράφος», «κόμβος», «ακμή». Επομένως, ιδανική για την υλοποίησή της φαινόταν μια αντικειμενοστραφής γλώσσα. Επίσης, η εφαρμογή είναι σχετικά υψηλού επιπέδου, γεγονός που συντέλεσε στην επιλογή της Java έναντι των αντικειμενοστραφών γλωσσών της οικογένειας της C, καθώς δε φαινόταν απαραίτητη η χρήση χαρακτηριστικών, όπως οι δείκτες και η άμεση αποδέσμευση μνήμης, ενώ μπορεί να έκαναν τον κώδικα πιο δύσκολο να αναπτυχθεί και πιο επιρρεπή σε λάθη. Εκτός από τα παραπάνω, η Java είναι ιδανική για υλοποίηση γραφικού περιβάλλοντος και γραφικής διεπαφής με το χρήστη. Τέλος, το γεγονός ότι η γλώσσα είναι ανοιχτού κώδικα, η μεταφερσιμότητα και η αξιοπιστία του κώδικα που έχει γραφεί σε Java και η πληθώρα βιβλιοθηκών συνέβαλλαν στην επιλογή της.

5.1.2 Το περιβάλλον ανάπτυξης εφαρμογών Eclipse

Η εφαρμογή αναπτύχθηκε στο περιβάλλον Eclipse. Ακολουθεί μια σύντομη εισαγωγή στο Eclipse και οι λόγοι επιλογής του.

5.1.2.1 Λίγα λόγια για το Eclipse

Το Eclipse είναι μια πλατφόρμα ανάπτυξης εφαρμογών. Είναι βασισμένο σε Java, ανοιχτού κώδικα και επεκτάσιμο. Το Eclipse είναι ένα πλαίσιο εργασίας που αποτελείται ένα σύνολο υπηρεσιών που προσαρτήθηκαν και βοηθούν στην ανάπτυξη μιας εφαρμογής. Εκτός όμως από τις ήδη ενσωματωμένες υπηρεσίες, υπάρχουν πολλά συστατικά που μπορούν να προσαρτηθούν σ' αυτό για επιπλέον δυνατότητες, όπως βιβλιοθήκες για γραφική δημιουργία εφαρμογών με γραφικά ή εφαρμογών διαδικτύου.

Εκτός από τα ήδη διαθέσιμα εργαλεία, το Eclipse έχει ενσωματωμένο ένα περιβάλλον ανάπτυξης, για όσους επιθυμούν να το επεκτείνουν, καθώς επιτρέπει τη δημιουργία εργαλείων που μπορούν να ενοποιηθούν απόλυτα με το περιβάλλον του Eclipse. Επειδή τα πάντα στο Eclipse είναι διακριτά τμήματα που ενσωματώνονται, δίνεται η δυνατότητα σε όσους αναπτύσσουν εργαλεία να προσφέρουν επεκτάσεις στο Eclipse και να δημιουργήσουν ένα συνεπές και ολοκληρωμένο περιβάλλον για τους χρήστες.

Παρόλο που το Eclipse έχει γραφτεί στη γλώσσα προγραμματισμού Java, η χρήση του δεν περιορίζεται σ' αυτή τη γλώσσα. Υπάρχουν διαθέσιμα (ή σχεδιάζονται) τμήματα για προσάρτηση στο Eclipse για την ανάπτυξη εφαρμογών σε γλώσσες προγραμματισμού όπως οι C/C++ και η COBOL. Το πλαίσιο εργασίας Eclipse μπορεί να χρησιμοποιηθεί επίσης και για άλλους τύπους εφαρμογών που δε σχετίζονται με την ανάπτυξη λογισμικού, όπως για Συστήματα Διαχείρισης Περιεχομένου (Content Management Systems) [Gall02].

5.1.2.2 Λόγοι επιλογής του Eclipse

Όπως αναφέρθηκε και στην προηγούμενη παράγραφο, το Eclipse είναι ένα προϊόν ανοικτού κώδικα, είναι γραμμένο σε Java και προσφέρεται ιδιαίτερα για την ανάπτυξη εφαρμογών στη γλώσσα αυτή.

Το Eclipse προσφέρει μορφοποίηση του κώδικα, εισαγωγή έτοιμων τμημάτων κώδικα και εύρεση ορισμού ή αναφοράς όρου (ιδιαίτερα χρήσιμο με μεγάλες αλυσίδες κλήσεων συναρτήσεων). Σε κάθε βήμα υπάρχει δυνατότητα για εμφάνιση προτάσεων ολοκλήρωσης του βήματος, γεγονός που κάνει ευκολότερη και γρηγορότερη την ανάπτυξη του κώδικα.

Με το Eclipse δίνεται η δυνατότητα κατά τη διαδικασία ανάπτυξης της εφαρμογής για ταυτόχρονη επόπτευση:

- του κώδικα
- της κλάσης
- των πακέτων που απαρτίζουν την εφαρμογή
- τυχόν σφαλμάτων

Ειδικά όσον αφορά τα σφάλματα, επειδή στο Eclipse η μεταγλώττιση του κώδικα είναι συνεχής, τυχόν λάθη εντοπίζονται κατά τη συγγραφή και προτείνονται από το περιβάλλον τρόποι διόρθωσής τους.

Το περιβάλλον παρέχει επίσης φυλλομετρητή που επιτρέπει την εμφάνιση της δομής του κώδικα και των κλάσεων που τον απαρτίζουν, καθώς και αποσφαλματωτή για ευκολότερη εύρεση σφαλμάτων χρόνου εκτέλεσης [Lourid].

5.1.3 Η βιβλιοθήκη για οπτικοποίηση γράφων: JUNG

Για τη γραφική αναπαράσταση των γράφων χρησιμοποιήθηκε η βιβλιοθήκη JUNG (Java Universal Network/Graph Framework). Ακολουθεί μια εισαγωγή στο JUNG και στη συνέχεια οι λόγοι επιλογής του.

5.1.3.1 Λίγα λόγια για τη βιβλιοθήκη JUNG

Το JUNG είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα, η οποία παρέχει μια κοινή και επεκτάσιμη γλώσσα για μοντελοποίηση, ανάλυση και οπτικοποίηση δεδομένων, τα οποία μπορούν να αναπαρασταθούν ως γράφος ή δίκτυο. Είναι γραμμένη σε Java, γεγονός που επιτρέπει στις εφαρμογές που βασίζονται στο JUNG να κάνουν χρήση των εκτεταμένων ενσωματωμένων δυνατοτήτων των προγραμματιστικών διεπαφών (API) της Java , καθώς και άλλων βιβλιοθηκών Java.

Η αρχιτεκτονική του JUNG έχει σχεδιαστεί έτσι, ώστε να υποστηρίζει μια ποικιλία αναπαραστάσεων οντοτήτων και των μεταξύ τους σχέσεων, όπως κατευθυνόμενους και μη-κατευθυνόμενους γράφους, πολύ-τροπικούς γράφους, γράφους με παράλληλες πλευρές και υπεργράφους. Παρέχει επίσης ένα μηχανισμό για την σήμανση γράφων, οντοτήτων και σχέσεων με μεταδεδομένα. Αυτό διευκολύνει τη δημιουργία εργαλείων για ανάλυση πολύπλοκων συνόλων δεδομένων, τα οποία μπορούν να εξετάσουν τις σχέσεις μεταξύ των οντοτήτων, καθώς και τα μεταδεδομένα κάθε οντότητας και σχέσης.

Η τρέχουσα έκδοση του JUNG περιλαμβάνει υλοποιήσεις αλγορίθμων θεωρίας γράφων, εξόρυξης δεδομένων και κοινωνικών δικτύων ανάλυσης, όπως ρουτίνες για ομαδοποίηση, αποσύνθεση, βελτιστοποίηση, δημιουργία τυχαίων γράφων κ.λ.π.

Το JUNG προσφέρει επίσης ένα πλαίσιο εργασίας για οπτικοποίηση, το οποίο διευκολύνει την κατασκευή εργαλείων για την εξερεύνηση των δεδομένων του γράφου με αλληλεπίδραση. Οι χρήστες μπορούν να κάνουν χρήση ενός από τους αλγορίθμους που παρέχονται για το πλάνο στο οποίο εμφανίζεται ο γράφος, ή να χρησιμοποιήσουν το πλαίσιο εργασίας για να δημιουργήσουν δικό τους πλάνο. Επιπλέον παρέχονται μηχανισμοί για φιλτράρισμα, οι οποίοι επιτρέπουν στους χρήστες να εστιάσουν την προσοχή τους ή την εφαρμογή των αλγορίθμων σε συγκεκριμένα τμήματα του γράφου.

Βασικές ιδιότητες και λειτουργίες

Οι γράφοι, οι κόμβοι και οι ακμές έχουν ορισμένες ιδιότητες που μπορούν να εξαχθούν απ' αυτά και λειτουργίες που μπορούν να εφαρμοστούν ή έχουν εφαρμοστεί πάνω σ' αυτά.

Διεπαφές

Οι διασυνδέσεις ArchetypeGraph, ArchetypeVertex και ArchetypeEdge καθορίζουν τη συμπεριφορά των γενικευμένων γράφων, κόμβων και ακμών. Έχουν σχεδιαστεί ώστε να περιγράφουν όλους τους τύπους γράφων, συμπεριλαμβανομένων των κατευθυνόμενων και των μη κατευθυνόμενων γράφων, των γράφων με προσαρτημένα δεδομένα (όπως ακμές με βάρη), των υπεργράφων και των γράφων με παράλληλες ακμές. Όλες οι υλοποιήσεις γράφων, κόμβων και ακμών πρέπει να υλοποιούν την κατάλληλη από αυτές τις διασυνδέσεις (ή μια

διασύνδεση που κληρονομεί από αυτές τις διασυνδέσεις). Οι μέθοδοι που παρατίθενται παραπάνω είναι διαθέσιμες στα αντικείμενα που υλοποιούν μια από αυτές τις διασυνδέσεις.

Οι διασυνδέσεις Graph, Vertex και Edge κληρονομούν από τις διασυνδέσεις Archetype και καθορίζουν τη συμπεριφορά των δυαδικών γράφων, δηλαδή των γράφων στους οποίους κάθε ακμή συνδέει ακριβώς δύο κόμβους. Αυτή η προδιαγραφή επιτρέπει τον ορισμό ενός αριθμού επιπλέον μεθόδων.

Οι διασυνδέσεις DirectedGraph και DirectedEdge καθορίζουν τη συμπεριφορά και τις δυνατότητες των κατευθυνόμενων γράφων και ακμών. Η DirectedEdge είναι ένας τύπος ακμής που επιβάλλει μια διάταξή στους κόμβους που την ορίζουν. Ο DirectedGraph είναι μια διασύνδεση για υλοποιήσεις γράφων, το σύνολο των ακμών των οποίων αποτελείται από υλοποιήσεις DirectedEdge.

Οι διασυνδέσεις UndirectedGraph και UdirectedEdge είναι οι αντίστοιχες των διασυνδέσεων για τους μη κατευθυνόμενους γράφους και τις μη κατευθυνόμενες ακμές.

Αφηρημένες κλάσεις

Οι κλάσεις AbstractSparseGraph, AbstractSparseVertex και AbstractSparseEdge σχεδιάστηκαν για αραιούς γράφους. Μπορεί να μην είναι οι καλύτερες υλοποιήσεις για το χειρισμό πυκνών γράφων.

Υλοποιημένες κλάσεις

Οι κλάσεις DirectedSparse{Graph, Vertex, Edge} και UndirectedSparse{Graph, Vertex, Edge} κληρονομούν από τις Abstract κλάσεις για τη δημιουργία αυστηρώς κατευθυνόμενων και μη κατευθυνόμενων γράφων.

5.1.3.2 Λόγοι επιλογής του JUNG για την οπτικοποίηση του γράφου

Έπειτα από αναζήτηση και σε άλλα εργαλεία οπτικοποίησης γράφων σε Java, επιλέχθηκε το JUNG επειδή συγκεντρώνει τα παρακάτω πλεονεκτήματα, τα οποία δεν υπάρχουν όλα μαζί στα άλλα εργαλεία που εξετάσαμε:

- Είναι ανοιχτού κώδικα, όπως και η Java, και εκμεταλλεύεται τις προγραμματιστικές διασυνδέσεις της.
- **Δεν** είναι απλώς ένα εργαλείο για οπτικοποίηση: επιτρέπει το γραφικό χειρισμό του γράφου.
- Παρέχει αφηρημένες κλάσεις, διασυνδέσεις αλλά και υλοποιημένες κλάσεις που κάνουν πιο εύκολη την αναπαράσταση των εννοιών που χρειαζόμαστε.
- Επιτρέπει την επέκταση των ήδη υπαρχόντων κλάσεων, ώστε να συμπεριλάβουμε επιπλέον μεταβλητές και μεθόδους που χρειαζόμαστε.

- Αποτρέπει τη δημιουργία κόμβων/ακμών που δε συνδέονται σε ένα γράφο (δηλαδή, δεν είναι μέρη ενός γράφου). Μειώνει έτσι την πιθανότητα σφαλμάτων.
- Παρέχει υλοποιημένους αλγορίθμους για γράφους, όπως αναφέρθηκε στην προηγούμενη υποενότητα.
- Παρέχει ποικιλία στον τρόπο αναπαράστασης των στοιχείων του γράφου (διαφορετικά χρώματα και σχήματα σε κόμβους/ακμές, εμφάνιση ονομάτων δίπλα από τους κόμβους/ακμές).
- Περιλαμβάνει παραδείγματα κώδικα που βοηθούν στην κατανόηση της λειτουργίας και των δυνατοτήτων του.

Γενικώς, το JUNG προσφέρει ευκολία στη χρήση, προγραμματιστική ευελιξία και αξιοπιστία [MFWB].

5.2 Εγκατάσταση του συστήματος.

Το σύστημα λειτουργεί με jre (Java Runtime Environment) 1.6. Για τη χρήση των πακέτων HecataeusJungGraph και Hecataeus, τα οποία χρειάζονται για την οπτικοποίηση του γράφου και το γραφικό χειρισμό του, απαιτείται η εισαγωγή των παρακάτω βιβλιοθηκών του JUNG (Java Universal Network/Graph Framework):

- colt (release 1.2)
- jung (jung 1.7.6)
- jung2-alpha (jung2-alpha2)
- jung-jai (jung-jai-1.0)
- jung-prefuse (alpha)

Ο χρήστης μπορεί να βρει τις παραπάνω βιβλιοθήκες και να τις πάρει δωρεάν από την ιστοσελίδα: SourceForge.net (http://sourceforge.net/project/showfiles.php?group_id=73840).

6 Έλεγχος

Στην ενότητα αυτή παρουσιάζεται αναλυτικά στο χρήστη ο τρόπος χρήσης της εφαρμογής και παράλληλα γίνεται η αξιολόγηση του συστήματος.

6.1 Μεθοδολογία ελέγχου

Ο έλεγχος του συστήματος θα γίνει με τη χρήση ενός σεναρίου λειτουργίας. Συγκεκριμένα, θα εισαχθεί στο σύστημα, με τη μορφή αρχείου xml, ο γράφος που αναπαριστά το σχήμα μιας βάσης δεδομένων και τα ερωτήματα/όψεις που την προσπελαύνουν.

Η βάση που θα χρησιμοποιηθεί περιλαμβάνει δύο σχέσεις, την TEST και την TESTB. Τη βάση προσπελαύνει μια όψη, η TEST_V και τέσσερα ερωτήματα. Το DDL αρχείο των πινάκων που θα εισαχθούν, καθώς το SQL αρχείο της όψης και των ερωτημάτων που προσπελαύνουν τους πίνακες παρουσιάζονται στο Παράρτημα (Παράγραφος 8.1).

Αφού εισαχθεί ο γράφος παρουσιάζεται ο γραφικός χειρισμός του γράφου που έχει οπτικοποιηθεί στο πλάνο. Ακολουθεί η προσθήκη σημασιολογίας στους κόμβους του γράφου που αφορά στην εξέλιξη του σχήματος της βάσης δεδομένων. Έπειτα, βάσει των ορισμένων πολιτικών και ενός συγκεκριμένου γεγονότος στο σχήμα της βάσης, βρίσκεται ο αντίκτυπος πάνω στο σύστημα της βάσης δεδομένων και των ερωτημάτων/όψεων. Τέλος, αποθηκεύονται τα αποτελέσματα των τροποποιήσεων που έγιναν πάνω στο γράφο.

6.2 Αναλυτική παρουσίαση ελέγχου

Στην παράγραφο αυτή παρουσιάζουμε αναλυτικά τον έλεγχο του συστήματος, σύμφωνα με το σενάριο που περιγράφηκε στην προηγούμενη ενότητα.

Το κεφάλαιο αυτό, λειτουργεί και ως εγχειρίδιο χρήσης του προγράμματος.

6.2.1 Εισαγωγή του σχήματος βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν– Χειρισμός του συνόλου του γράφου.

Το παράθυρο που βλέπει ο χρήστης ξεκινώντας την εφαρμογή είναι:

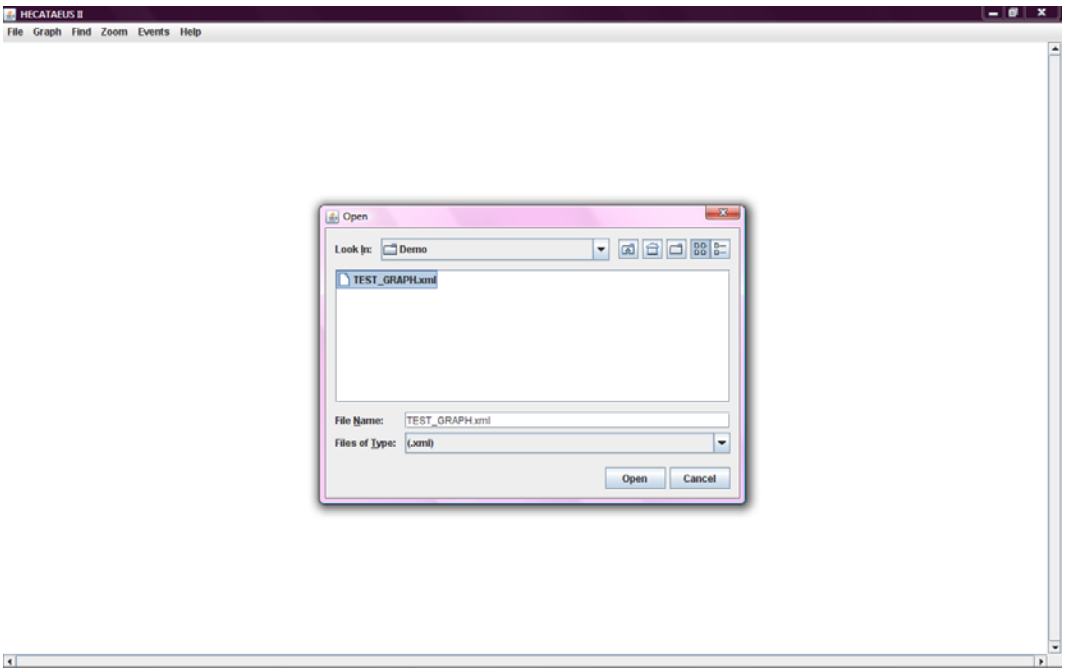
Εικόνα 6.1. Κυρίως παράθυρο της εφαρμογής

6.2.1.1 Εισαγωγή του γράφου.

Από εκεί, επιλέγοντας «Import graph» από το «File» του μενού επιλογών, μπορεί να εισάγει έναν γράφο που αναπαριστά μια βάση δεδομένων και τα ερωτήματα προς αυτή.

Εικόνα 6.2. Εισαγωγή γράφου από αρχείο xml.

Το αρχείο από το οποίο θα εισαχθεί ο γράφος επιλέγεται από το παράθυρο που εμφανίζεται, αφού ο χρήστης πατήσει «Import graph», στο οποίο εμφανίζονται όλες οι μονάδες και οι φάκελοι που περιέχουν αρχεία τύπου xml, όπως φαίνεται στην Εικόνα 6.3.



Εικόνα 6.3. Επιλογή του αρχείου xml.

Ανοίγοντας το επιλεγμένο αρχείο εμφανίζεται ο γράφος στο πλάνο.

Αριστερά στο πλάνο απεικονίζονται τα ερωτήματα με τα γνωρίσματά τους, στο κέντρο απεικονίζονται οι όψεις με τα γνωρίσματά τους και δεξιά στο πλάνο οι σχέσεις με τα γνωρίσματά τους.

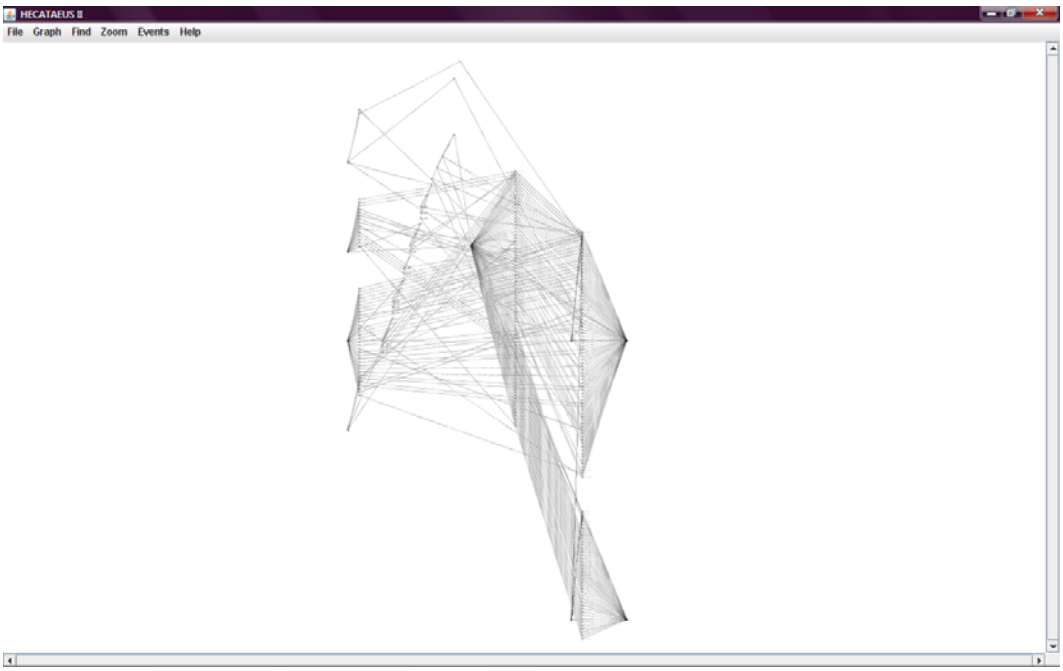
(Ο χρήστης έχει τη δυνατότητα να προσθέσει στο πλάνο υπογράφους, αυτό όμως θα παρουσιαστεί στην παράγραφο 6.2.1.4, αφού προηγηθεί η παρουσίαση των λειτουργιών μεγέθυνσης-σμίκρυνσης και των καταστάσεων λειτουργίας του ποντικιού.)

6.2.1.2 Λειτουργίες μεγέθυνσης-σμίκρυνσης

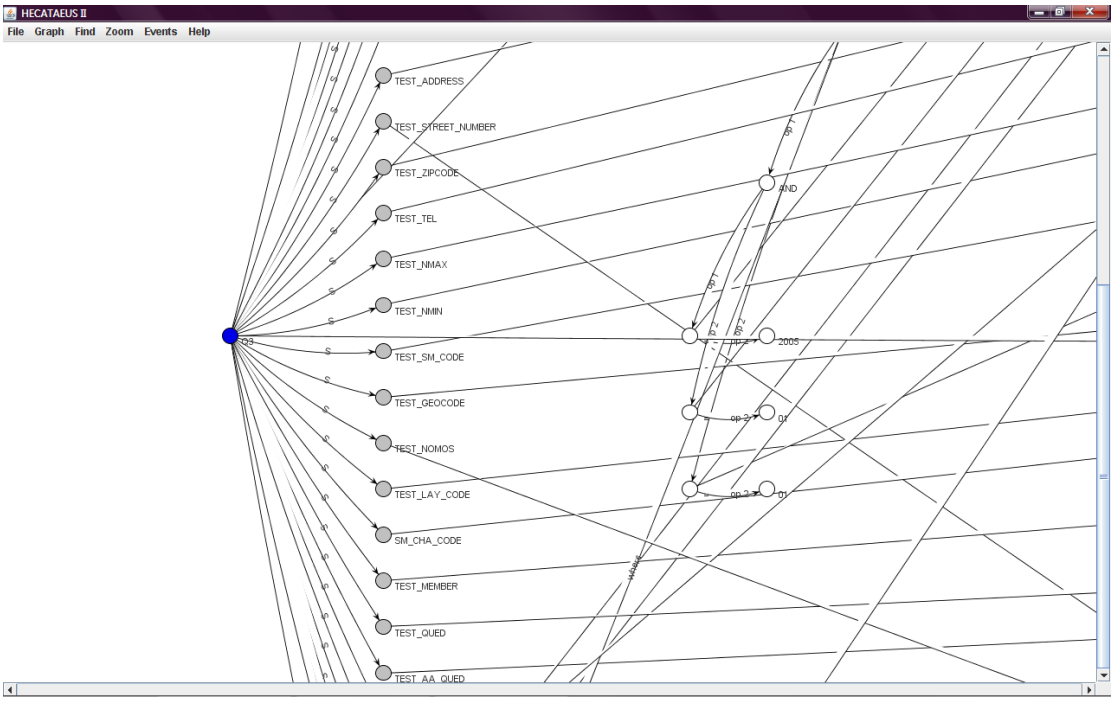
Επιλέγοντας «Zoom in»/«Zoom out» από το «Zoom» του μενού επιλογών ο χρήστης μπορεί να δει το γράφο με μεγαλύτερη ή μικρότερη λεπτομέρεια.

Εναλλακτικός τρόπος για αλλαγή της κλίμακας δίνεται με κύλιση της ροδέλας του ποντικιού.

Παραδείγματα απεικόνισης του συνολικού γράφου και ενός τμήματος του γράφου που δείχνει το ερώτημα Q3 και μερικά γνωρίσματά του φαίνονται στις Εικόνες 6.4 και 6.5, αντίστοιχα.



Εικόνα 6.4. Ο γράφος που εισήχθη έπειτα από σμίκρυνση.



Εικόνα 6.5. Ο γράφος που εισήχθη έπειτα από μεγέθυνση.

6.2.1.3 Καταστάσεις λειτουργιών ποντικιού

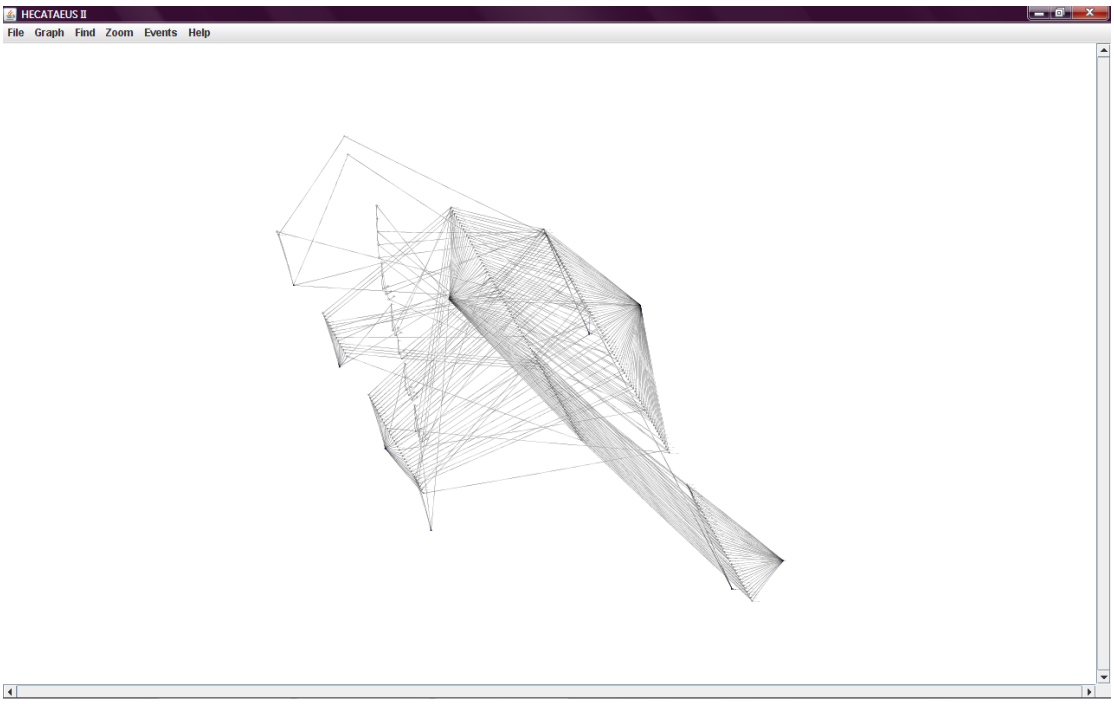
Υπάρχουν δύο καταστάσεις λειτουργιών του ποντικιού για ευκολότερο χειρισμό του γράφου, μια από τις οποίες μπορεί να επιλέγει ο χρήστης κατά τη διάρκεια εκτέλεσης της εφαρμογής από την επιλογή «Graph» του μενού επιλογών:

- Κατάσταση κίνησης που επιτυγχάνεται επιλέγοντας «Move».
- Κατάσταση επιλογής που επιτυγχάνεται επιλέγοντας «Pick».

Συνεπώς, αν ο χρήστης επιθυμεί να μετακινήσει το γράφο μέσα στο πλάνο για να δει άλλο τμήμα του, μπορεί να το κάνει επιλέγοντας «Move» από το «Graph» του μενού επιλογών, πατώντας το ποντίκι και σέρνοντας το δείκτη του προς την κατεύθυνση που θέλει να μετακινηθεί ο γράφος.

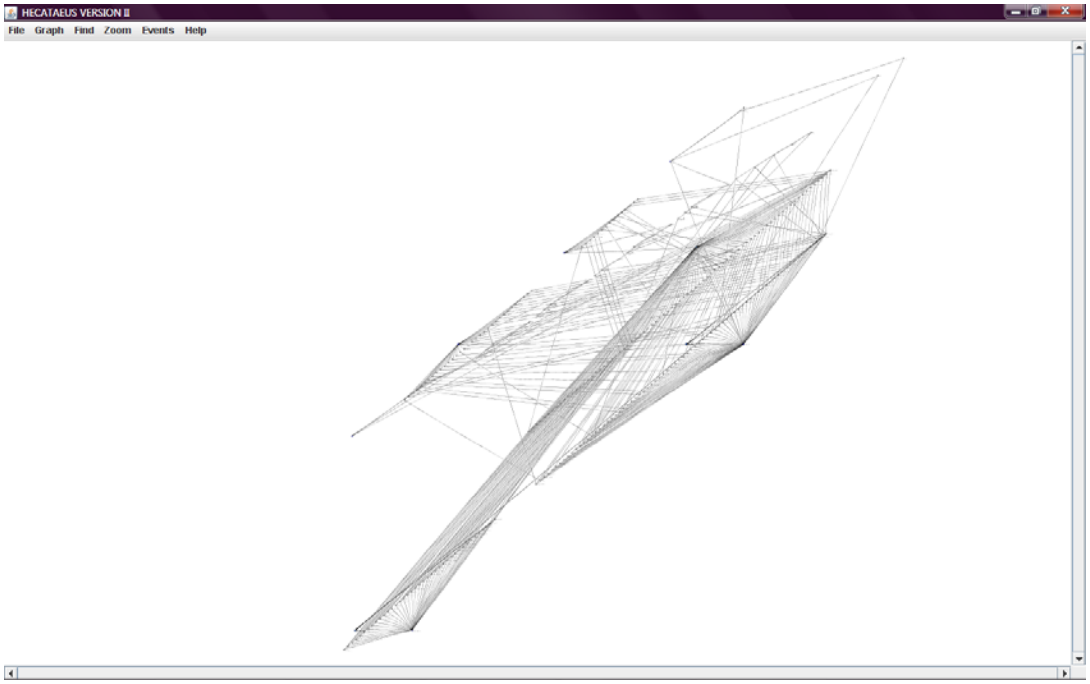
Έστω ότι ο χρήστης επιθυμεί να δει τα γνωρίσματα του ερωτήματος Q3 (της Εικόνας 6.5) που βρίσκονται πιο ψηλά στο πλάνο. Πατώντας και σέρνοντας το δείκτη του ποντικιού προς τα κάτω μετακινεί το γράφο προς τα κάτω.

Επίσης, στην κατάσταση κίνησης, μπορεί ο χρήστης να περιστρέψει το γράφο με το πάτημα του ποντικιού σε ένα σημείο, κρατώντας πατημένο το ποντίκι, το πλήκτρο «Shift» και σέρνοντας το δείκτη του ποντικιού:



Εικόνα 6.6. Ο γράφος της Εικόνας 6.4 μετά την περιστροφή προς τα αριστερά.

Στην κατάσταση κίνησης δίνεται, ακόμα, στο χρήστη η δυνατότητα παραμόρφωσης του γράφου, με το πάτημα του ποντικιού σε ένα σημείο, κρατώντας πατημένο το ποντίκι, το πλήκτρο «Ctrl» και σέρνοντας το δείκτη του ποντικιού προς την κατεύθυνση που επιθυμεί να επιμηκυνθεί ο γράφος. Για παράδειγμα, με πάτημα του ποντικιού σε ένα σημείο στο πάνω τμήμα του γράφου που φαίνεται στην Εικόνα 6.4 και σέρνοντας το δείκτη του ποντικιού προς τα δεξιά, ο γράφος γίνεται όπως φαίνεται στην Εικόνα 6.7.

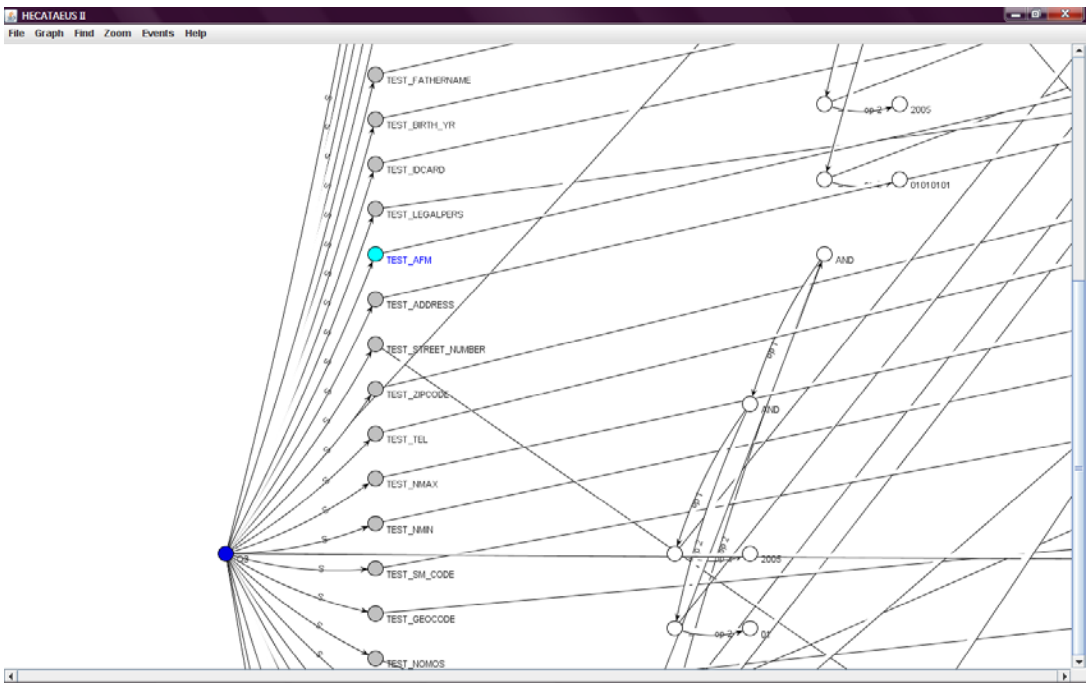


Εικόνα 6.7. Το αποτέλεσμα της παραμόρφωσης.

Επιλέγοντας «Pick» από το «Graph» του μενού επιλογών, ο χρήστης έχει τη δυνατότητα να επιλέξει ένα ή περισσότερα στοιχεία του γράφου (κόμβο ή ακμή), με πάτημα του ποντικιού πάνω σ' αυτό. Το χρώμα του επιλεγμένου στοιχείου γίνεται κυανό και η ονομασία του μπλε.

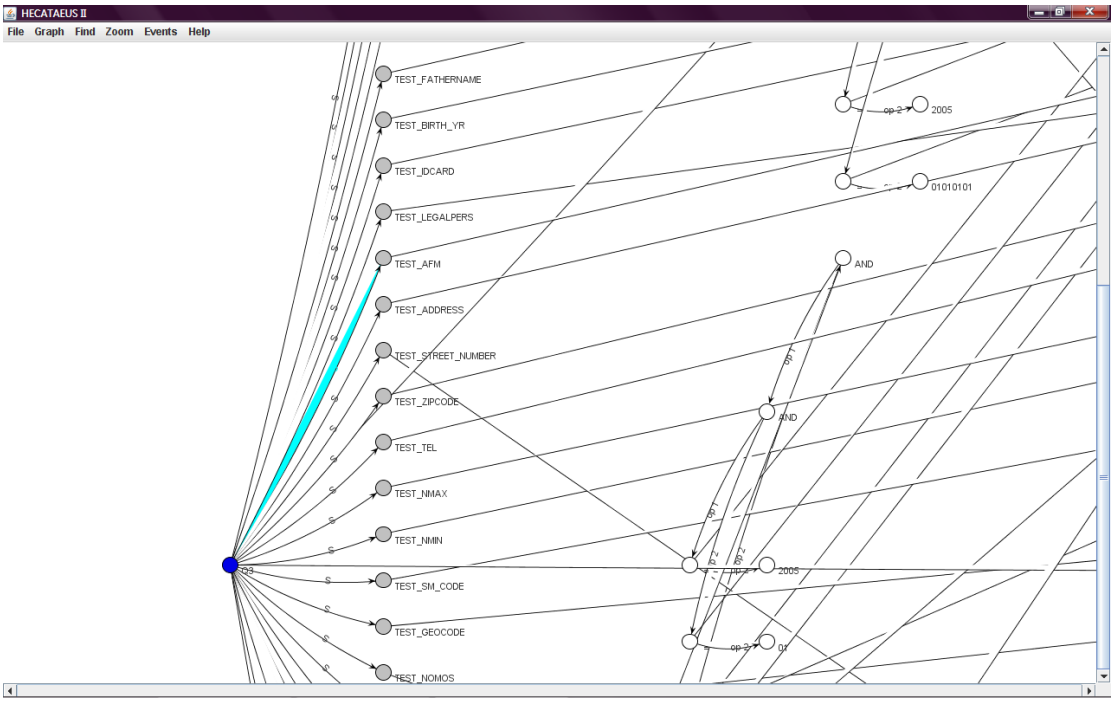
Τα στοιχεία του γράφου παύουν να είναι επιλεγμένα όταν ο χρήστης πατήσει το ποντίκι οπουδήποτε μέσα στο πλάνο.

- Παράδειγμα επιλεγμένου κόμβου:



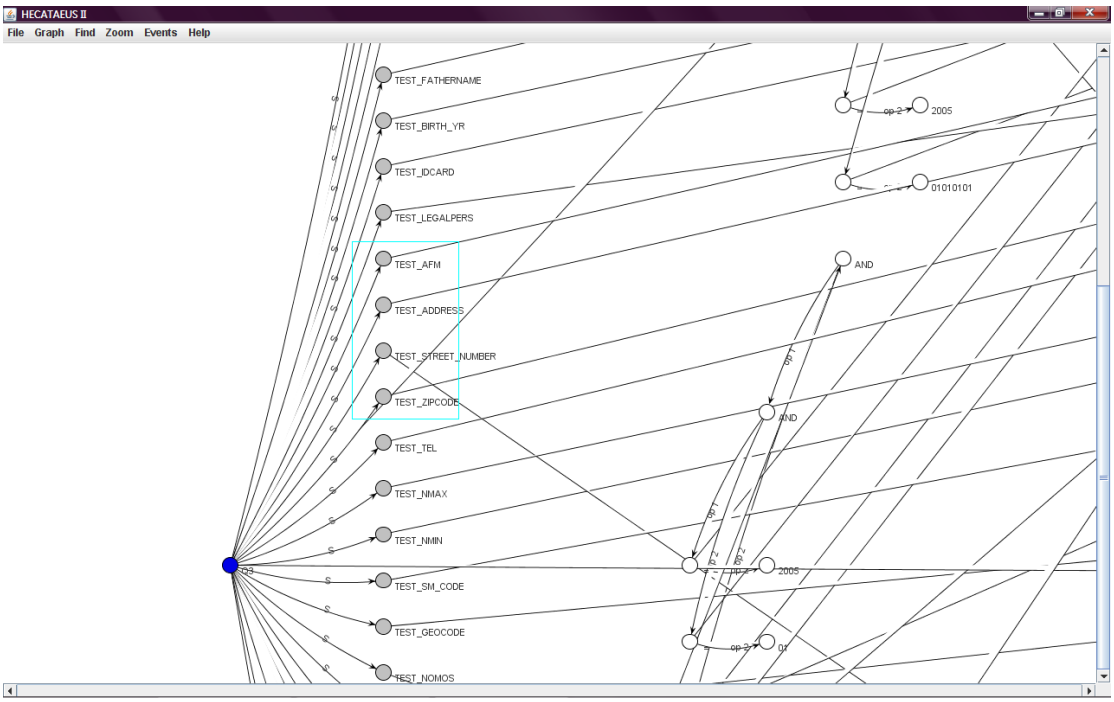
Εικόνα 6.8. Παράδειγμα επιλεγμένου κόμβου.

- Παράδειγμα επιλεγμένης ακμής:



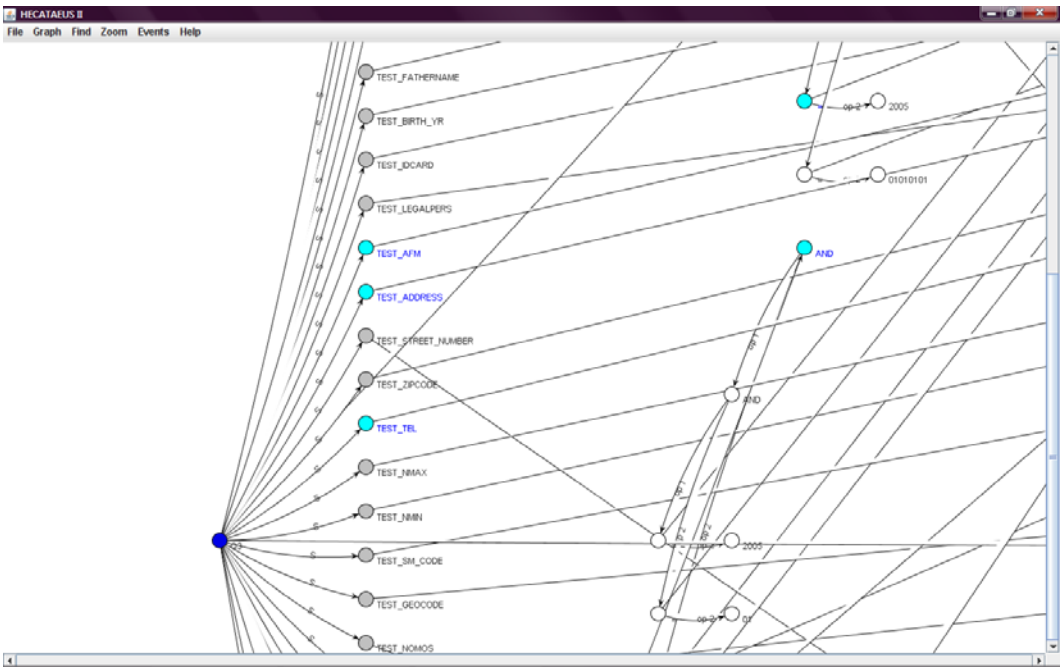
Εικόνα 6.9. Παράδειγμα επιλεγμένης ακμής.

Επίσης, μπορεί να επιλέξει περισσότερους από έναν κόμβους πατώντας και σέρνοντας το ποντίκι, ώστε να οι κόμβοι να περιέχονται εντός του ορθογωνίου που ορίζεται από την αρχική και την τελική θέση του δείκτη του ποντικιού, όπως φαίνεται στην Εικόνα 6.10.



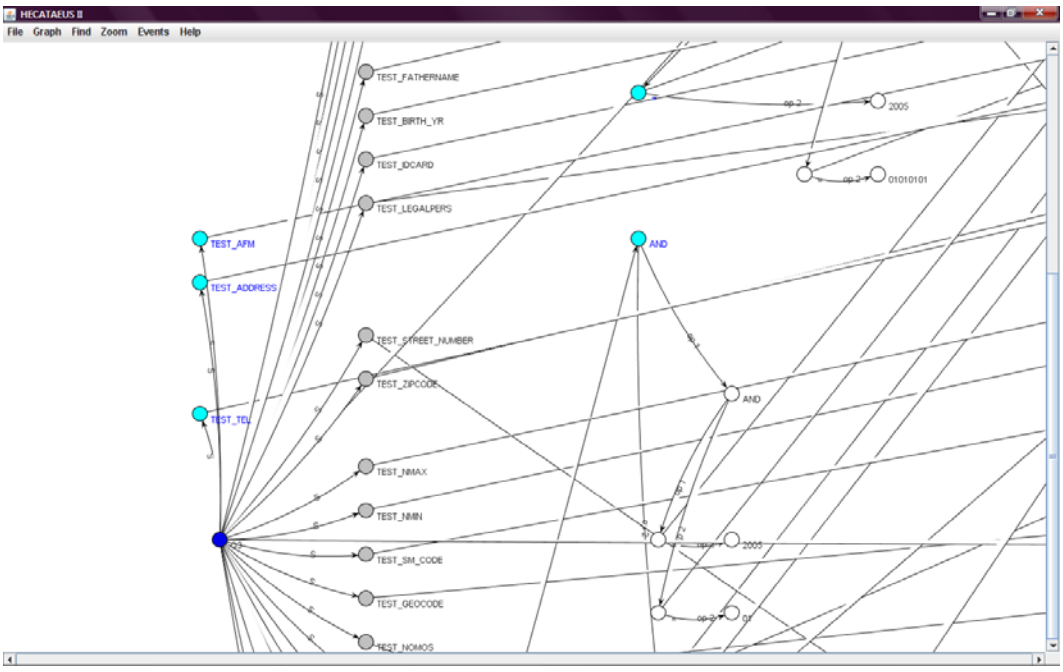
Εικόνα 6.10. Επιλογή πολλών κόμβων.

Εναλλακτικά, μπορεί να επιλέξει περισσότερους του ενός κόμβους κρατώντας το πλήκτρο "Shift" πατημένο και πατώντας το ποντίκι πάνω στους κόμβους. Μπορεί, κατά αυτόν τον τρόπο, να επιλέξει μη «διαδοχικούς» κόμβους, όπως φαίνεται στην Εικόνα 6.11.



Εικόνα 6.11. Παράδειγμα επιλεγμένων (μη διαδοχικών) κόμβων.

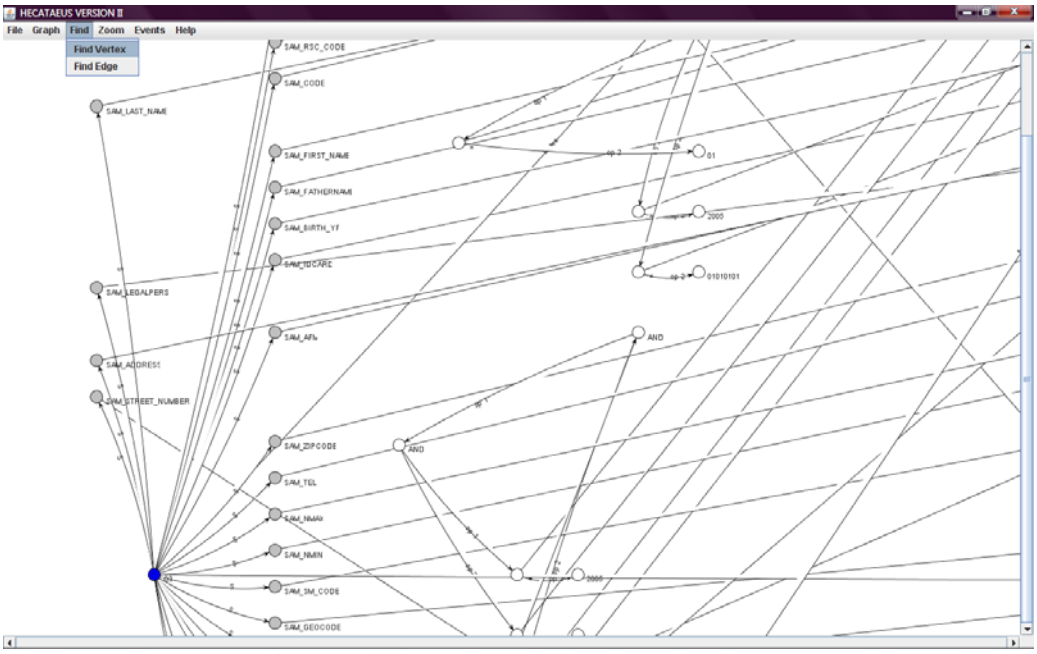
Με το πάτημα του ποντικιού πάνω σε έναν επιλεγμένο κόμβο, κρατώντας πατημένο το ποντίκι και σέρνοντας προς μια κατεύθυνση, όλοι οι επιλεγμένοι κόμβοι μετακινούνται προς την κατεύθυνση αυτή. Για παράδειγμα, πατώντας το ποντίκι και σέρνοντας προς τα αριστερά, οι επιλεγμένοι κόμβοι του γράφου μετακινούνται προς τα αριστερά.



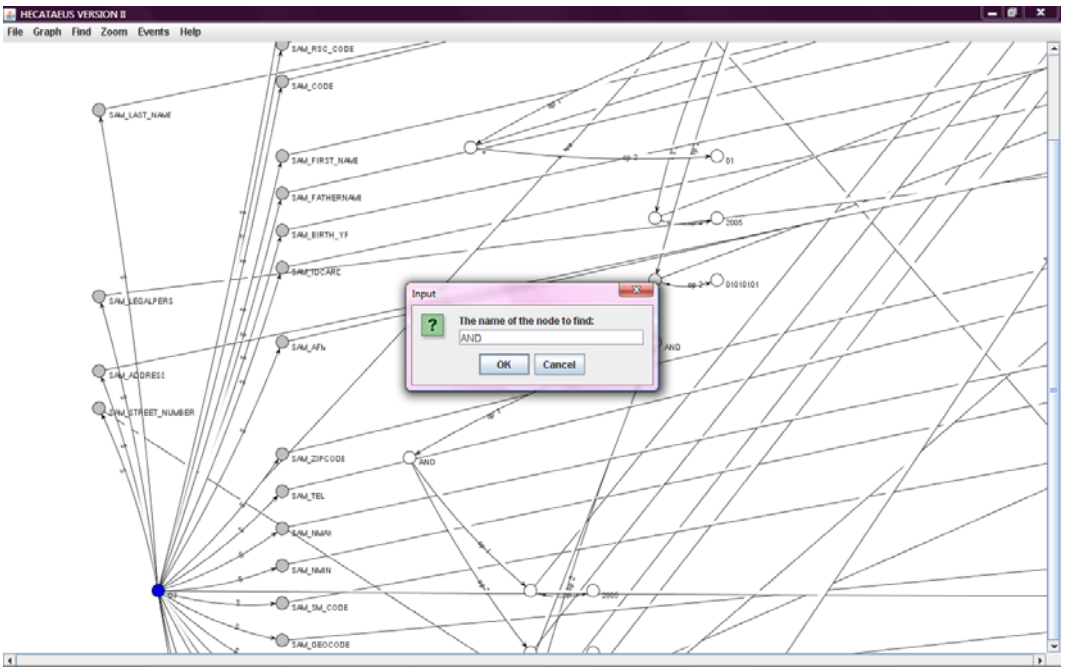
Εικόνα 6.12. Οι επιλεγμένοι κόμβοι της Εικόνας 6.11 μετακινημένοι προς τα αριστερά.

Ο ΕΚΑΤΑΙΟΣ ΙΙ δίνει ακόμα τη δυνατότητα στο χρήστη να βρει εύκολα έναν κόμβο ή μια ακμή στο πλάνο, απλά με το όνομά του.

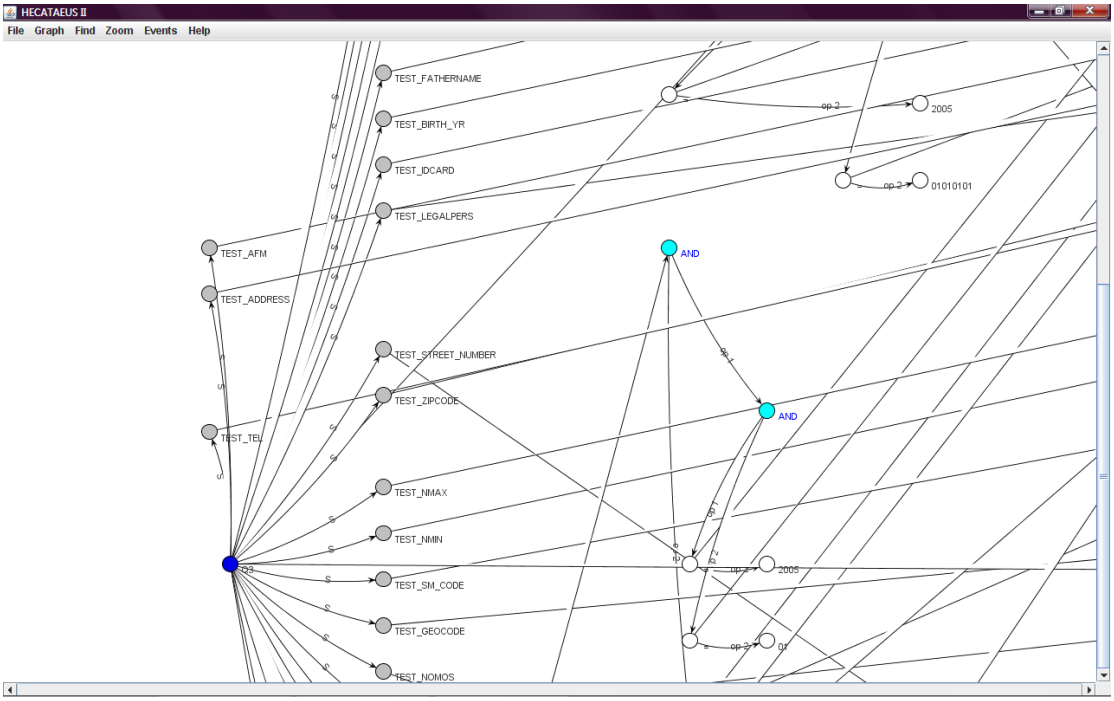
Έστω ότι ο χρήστης θέλει να βρει όλους τους κόμβους με όνομα «AND». Επιλέγοντας «Find Vertex» από το «Find» του μενού επιλογών (Εικόνα 6.13), εμφανίζεται ένα παράθυρο καθορισμού του ονόματος του κόμβου. Γράφοντας «AND» και πατώντας το κουμπί «OK» (Εικόνα 6.14), όλοι οι κόμβοι στο πλάνο με όνομα «AND» επιλέγονται, όπως φαίνεται στην Εικόνα 6.15.



Εικόνα 6.13. Εύρεση κόμβων με συγκεκριμένο όνομα.

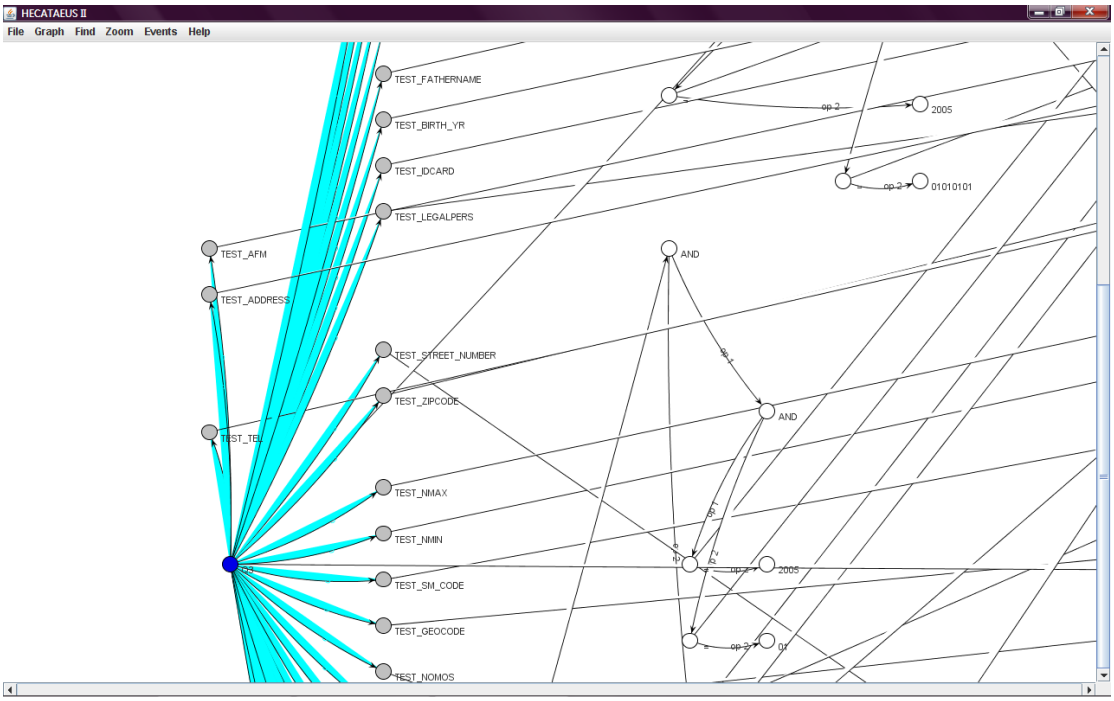


Εικόνα 6.14. Συμπλήρωση του πεδίου ονόματος για εύρεση των κόμβων με όνομα «AND».



Εικόνα 6.15. Οι κόμβοι με όνομα «AND» έχουν επιλεχτεί.

Με παρόμοιο τρόπο γίνεται η εύρεση ακμών με συγκεκριμένο όνομα. Για παράδειγμα, στην Εικόνα 6.16 φαίνονται οι ακμές με όνομα «S».



Εικόνα 6.16. Οι ακμές με όνομα «S» έχουν επιλεχτεί.

6.2.1.4 Εισαγωγή υπογράφου.

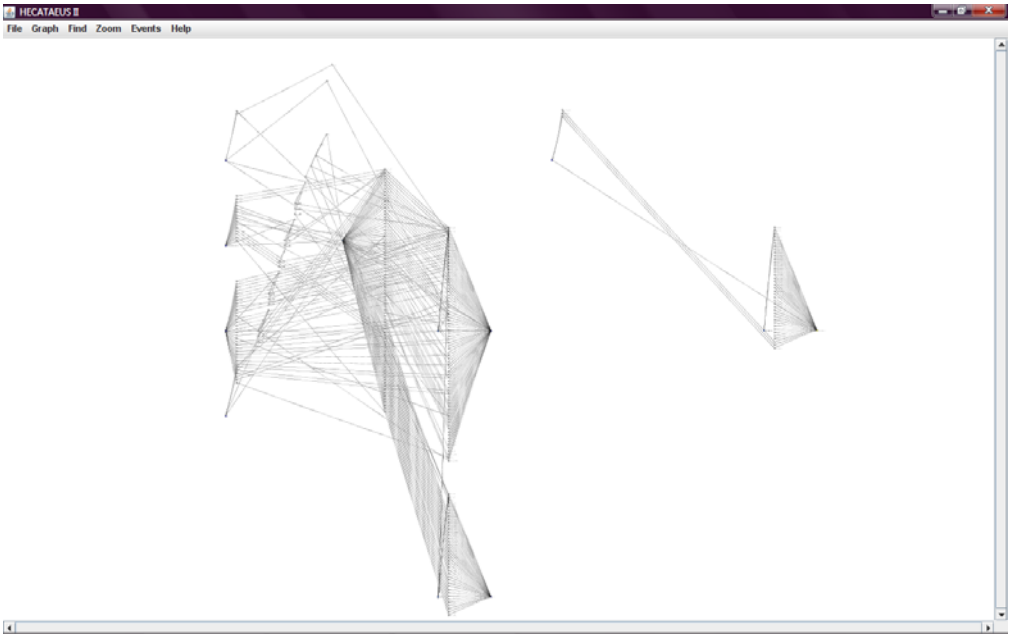
Όπως αναφέρθηκε και στην παράγραφο 6.2.1.1, ο χρήστης έχει τη δυνατότητα εισαγωγής υπογράφου, διατηρώντας και τον αρχικό γράφο. Αυτό γίνεται επιλέγοντας «Add Graph» από το «File» του μενού επιλογών.

Για το συγκεκριμένο σενάριο, έστω ότι ο χρήστης θέλει να προσθέσει έναν υπογράφο. (Είναι σημαντικό ο χρήστης να εισάγει τον υπογράφο όταν η λεπτομέρεια με την οποία φαίνεται ο γράφος είναι ίδια με την αρχική, ούτως ώστε οι δυο γράφοι να φαίνονται με την ίδια λεπτομέρεια.) Μετακινεί τον αρχικό γράφο προς τα δεξιά, ώστε να μη συμπέσει με τον υπογράφο και επιλέγει «Add Graph».

Εικόνα 6.17. Εισαγωγή υπογράφου.

Η επιλογή του αρχείου από το οποίο θα εισαχθεί ο γράφος γίνεται όπως και στην περίπτωση εισαγωγής του αρχικού γράφου, από ένα παράθυρο, στο οποίο εμφανίζονται όλες οι μονάδες και οι φάκελοι που περιέχουν αρχεία τύπου xml.

Έπειτα από την εισαγωγή του νέου γράφου με σμίκρυνση και μεγέθυνση φαίνονται στο πλάνο ο αρχικός γράφος και ο υπογράφος (Εικόνα 6.18).

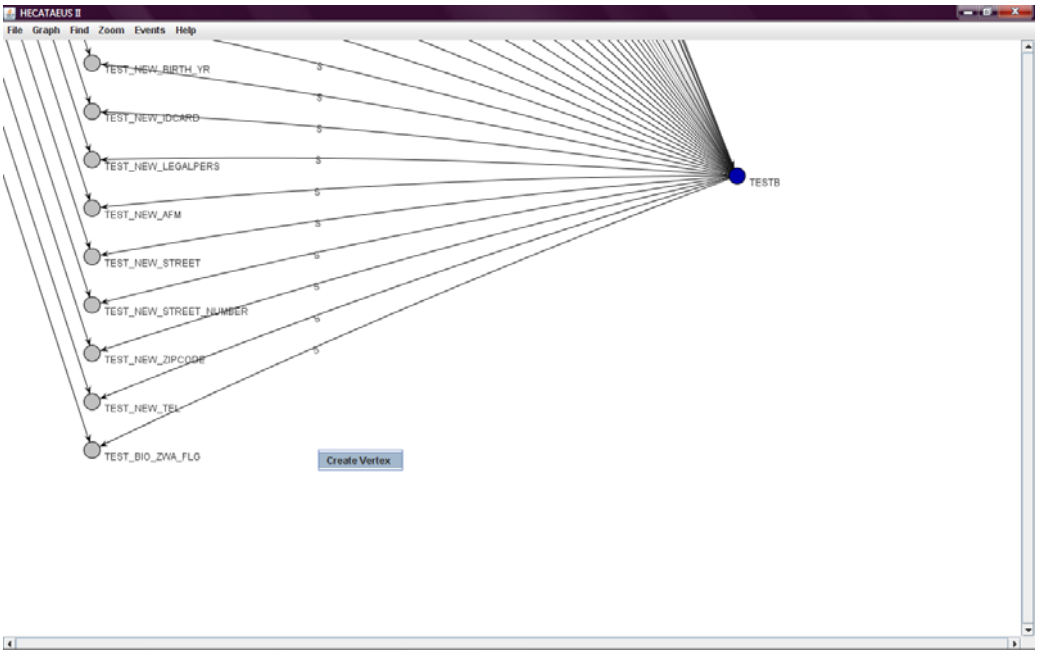


Εικόνα 6.18. Ο αρχικός γράφος και ο υπογράφος.

6.2.2 Χειρισμός των στοιχείων του γράφου

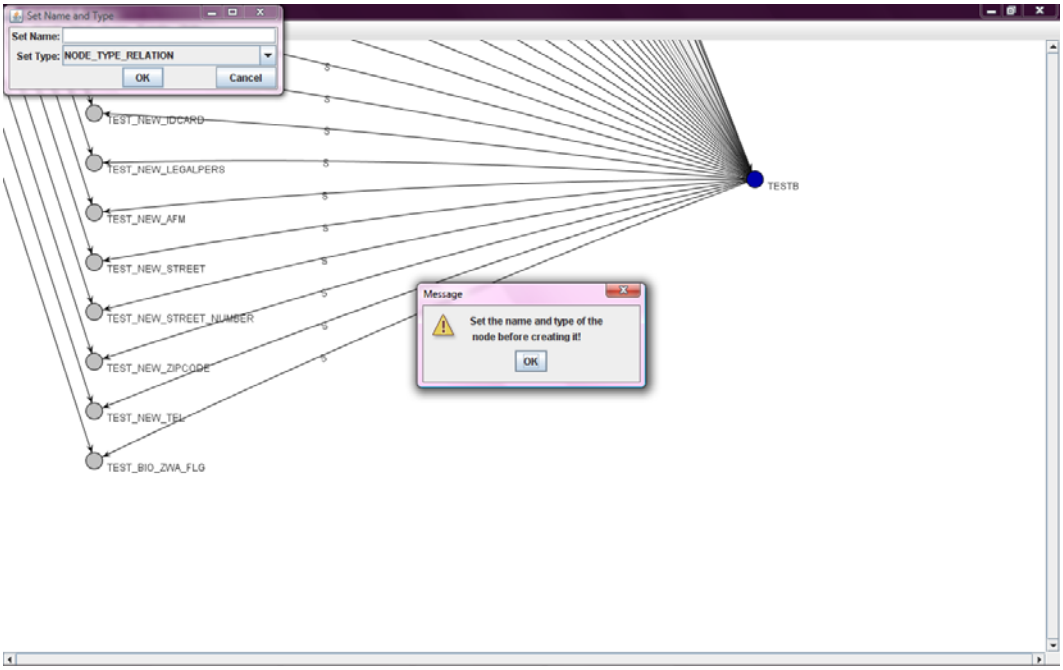
6.2.2.1 Δημιουργία κόμβων-δημιουργία ακμών-Χρωματισμός κόμβων.

Έστω τώρα ότι ο χρήστης θέλει να προσθέσει ένα νέο γνώρισμα στη σχέση TESTB. Μεγεθύνει και μετακινεί το γράφο όπως αναφέρθηκε παραπάνω για να φέρει τη σχέση TESTB στο πλάνο. Στη συνέχεια με δεξί πάτημα του ποντικιού σε ένα σημείο σε κενή περιοχή στο πλάνο (όχι πάνω σε κόμβο ή ακμή) εμφανίζεται ένα η επιλογή για δημιουργία νέου κόμβου στο σημείο αυτό (Εικόνα 6.19).



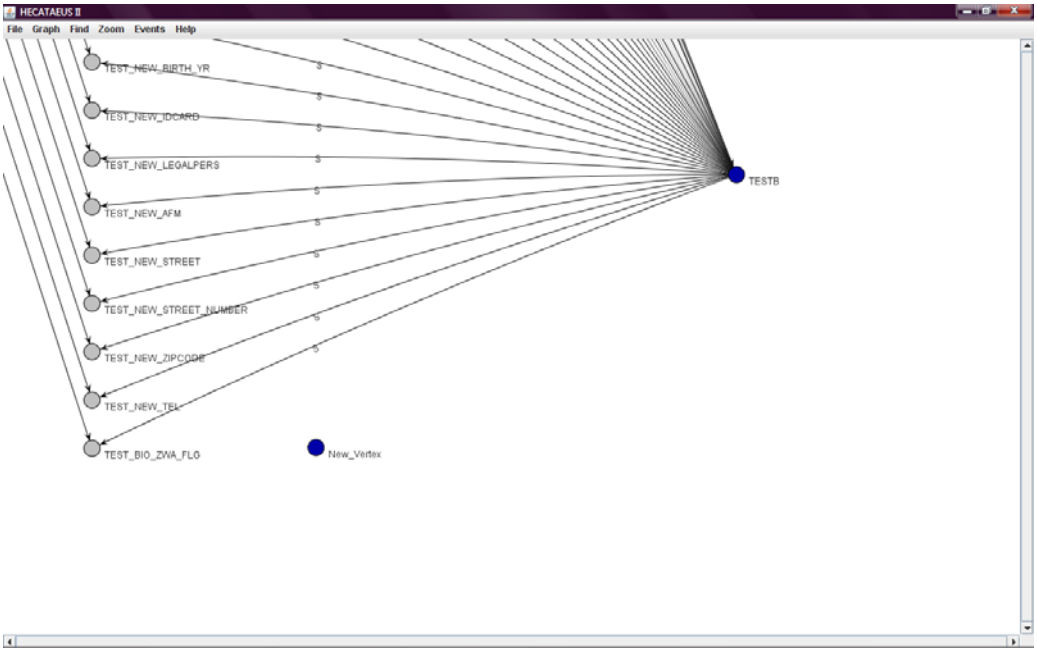
Εικόνα 6.19. Δημιουργία νέου κόμβου στη συγκεκριμένη θέση.

Πατώντας «Create Vertex» εμφανίζεται πάνω αριστερά στην οθόνη ένα παράθυρο στο οποίο συμπληρώνει το όνομα του κόμβου και επιλέγει έναν από τους καθορισμένους τύπους για κόμβο. Αν ο χρήστης δε συμπληρώσει το όνομα του κόμβου εμφανίζεται προειδοποιητικό μήνυμα και το σύστημα δε δημιουργεί τον κόμβο, μέχρι ο χρήστης να συμπληρώσει το πεδίο για το όνομα ή να ακυρώσει τη δημιουργία κόμβου.



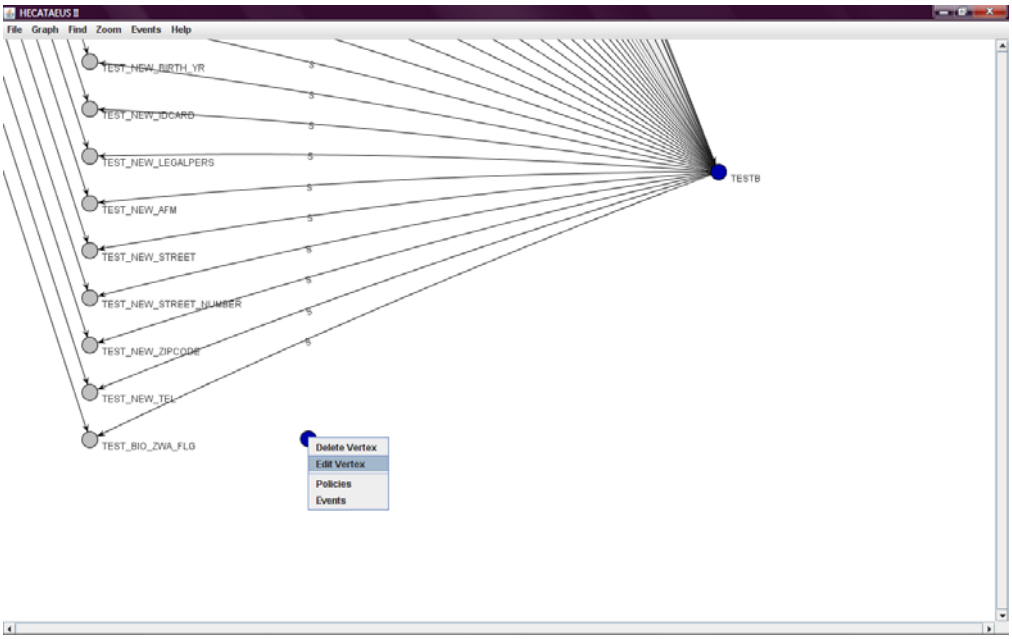
Εικόνα 6.20. Μήνυμα προειδοποίησης: Το πεδίο του ονόματος είναι κενό.

Γράφοντας «New_Vertex» στο πεδίο του ονόματος δημιουργείται ένας νέος κόμβος στο σημείο που πάτησε ο χρήστης το ποντίκι.



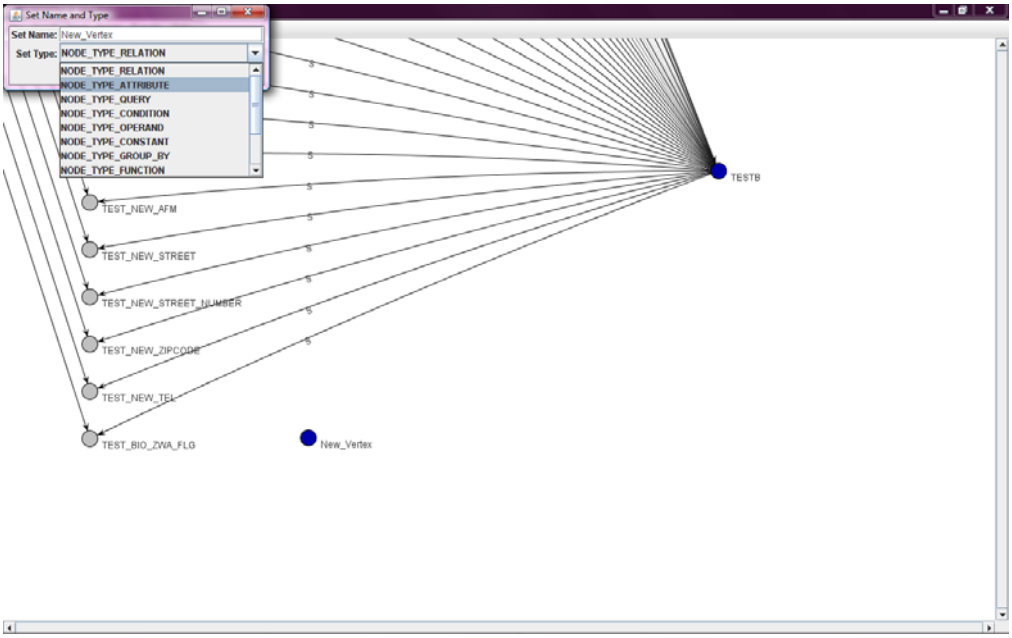
Εικόνα 6.21. Ο κόμβος New_Vertex έχει δημιουργηθεί.

Ο νέος κόμβος έχει όνομα «New_Vertex» και είναι τύπου «σχέση», εφόσον ο χρήστης δεν άλλαξε τον προεπιλεγμένο τύπο ακμής. Η αλλαγή των επιλογών που έχουν γίνει για έναν κόμβο γίνεται με δεξί πάτημα του ποντικιού στο συγκεκριμένο κόμβο. Για παράδειγμα, με δεξί πάτημα στον κόμβο New_Vertex, εμφανίζονται οι επιλογές:



Εικόνα 6.22. Εμφάνιση των ιδιοτήτων του κόμβου New_Vertex.

Επιλέγοντας «Edit Vertex» εμφανίζεται ξανά το παράθυρο ορισμού των ιδιοτήτων του κόμβου με τις επιλογές που είχαν γίνει, τις οποίες ο χρήστης μπορεί να αλλάξει. Για να αλλάξει τον τύπο του κόμβου σε «γνώρισμα» επιλέγει το συγκεκριμένο τύπο από τη λίστα επιλογών για τον τύπο του κόμβου.



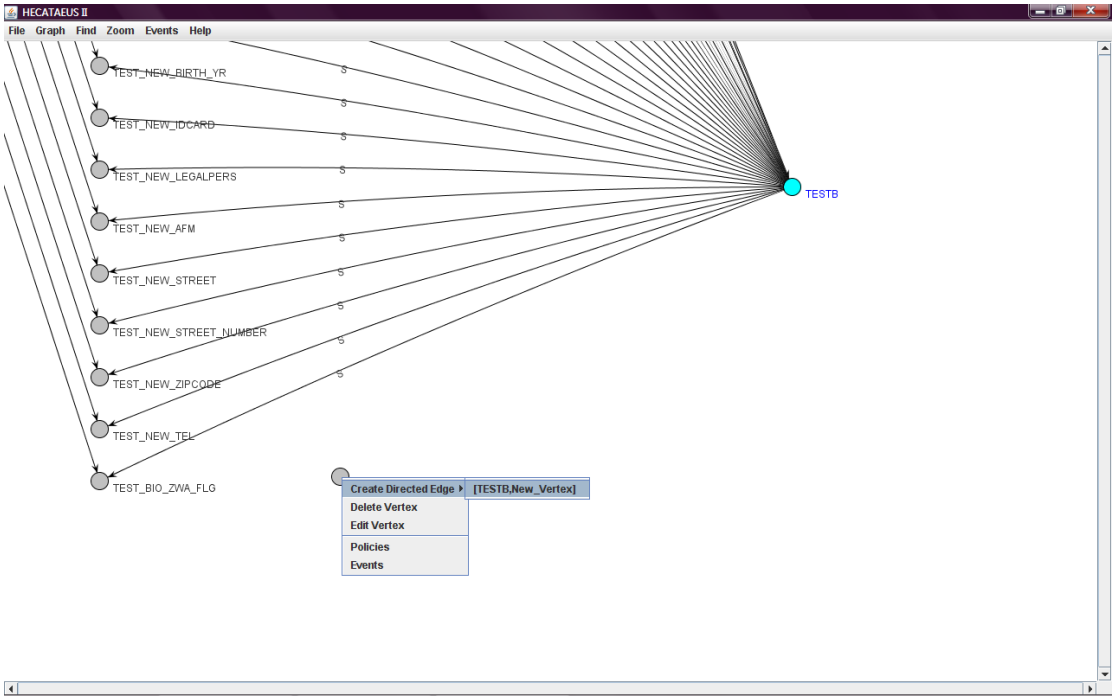
Εικόνα 6.23. Αλλαγή του τύπου του κόμβου New_Vertex σε «γνώρισμα».

Ο τύπος του κόμβου αλλάζει από «σχέση» σε «γνώρισμα» και το χρώμα του γράφου αλλάζει από σκούρο μπλε σε γκρι, αντανακλώντας οπτικά την αλλαγή που έγινε.

Ο χρωματισμός για κάθε τύπο κόμβου έχει ως εξής:

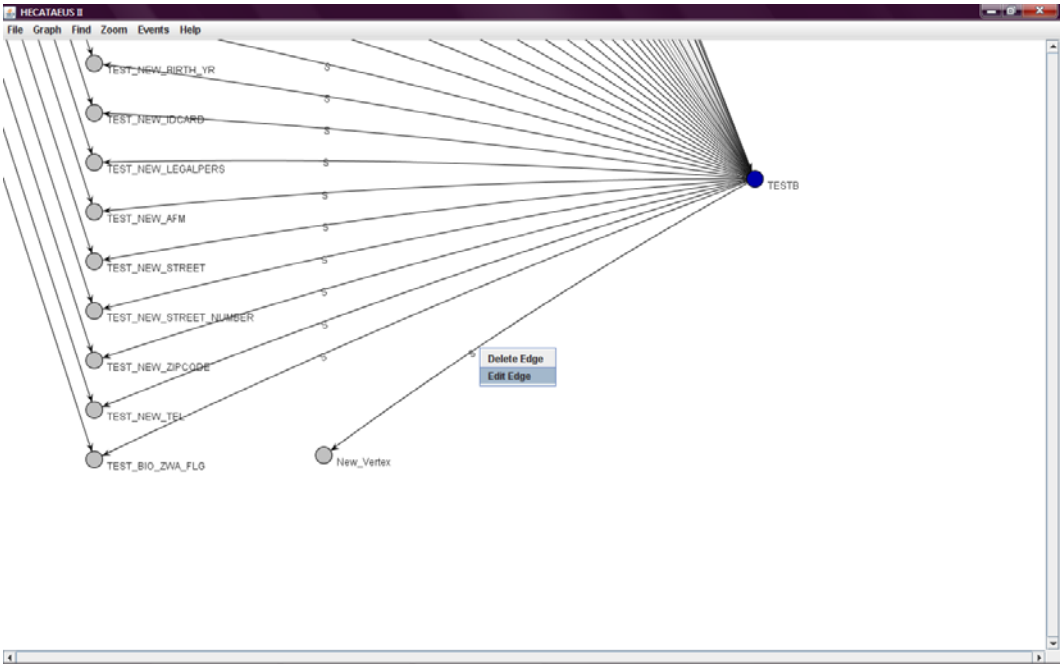
Εικόνα 6.24. Χρωματισμός κόμβων ανάλογα με τον τύπο τους.

Για να δημιουργήσει ο χρήστης την ακμή από τον κόμβο σχέση TESTB προς τον κόμβο γνώρισμα New_Vertex επιλέγει τον κόμβο TESTB και πατά το δεξί κουμπί του ποντικιού πάνω στον κόμβο New_Vertex. Από εκεί με την επιλογή «Create Directed Edge» του δίνεται η επιλογή δημιουργίας ακμής από τον κόμβο TESTB στον κόμβο New_Vertex.



Εικόνα 6.25. Δημιουργία νέας ακμής από τον κόμβο TESTB στον κόμβο New_Vertex.

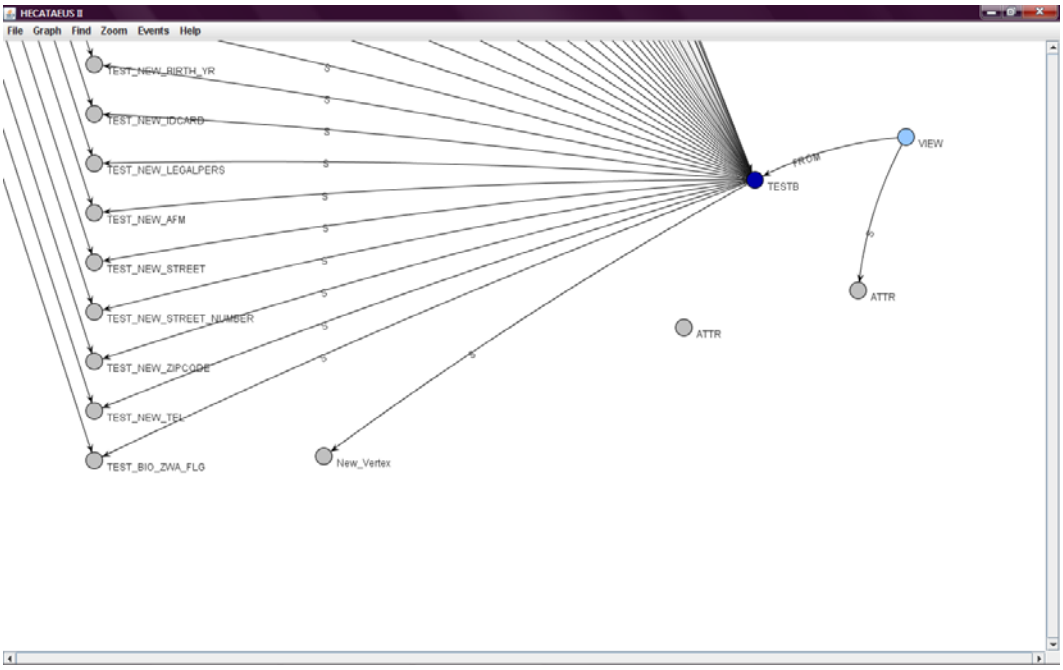
Οι ιδιότητες της ακμής μπορούν να αλλάξουν μετά τη δημιουργία της με παρόμοιο τρόπο με τον οποίο αλλάζουν οι ιδιότητες του κόμβου, επιλέγοντας «Edit Edge» και στη συνέχεια αλλάζοντας τις ιδιότητες από το παράθυρο, στο οποίο εμφανίζονται οι επιλογές που είχαν γίνει.



Εικόνα 6.26. Εμφάνιση/αλλαγή των ιδιοτήτων της ακμής.

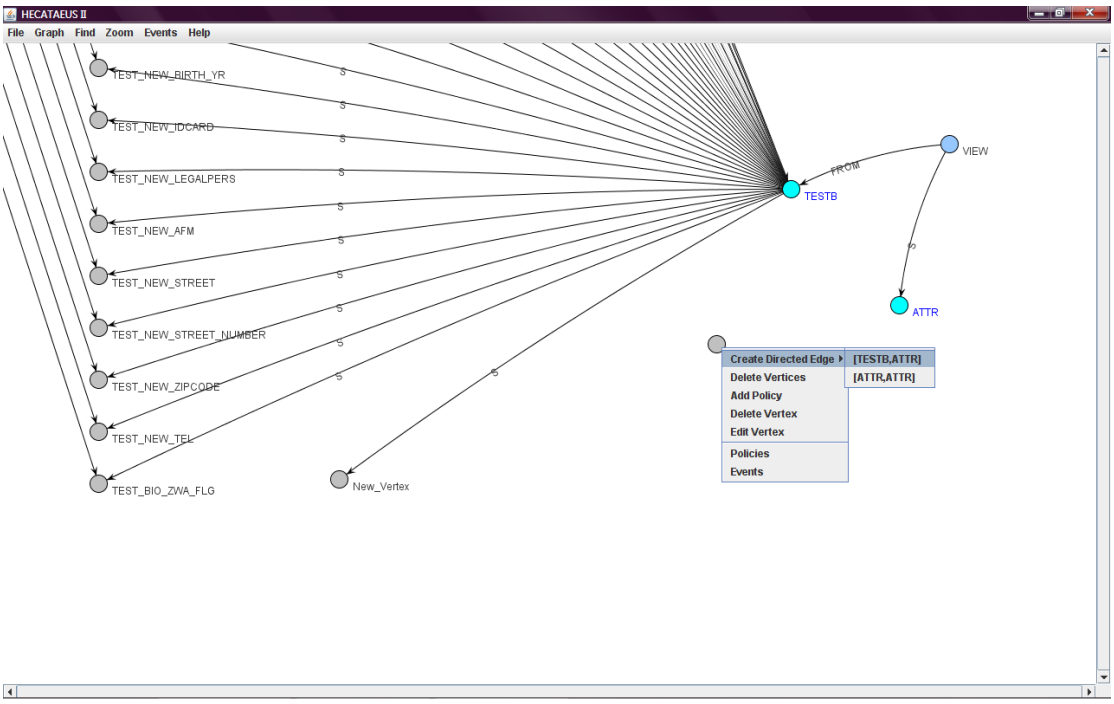
Δίνεται επίσης η δυνατότητα δημιουργίας ακμών από περισσότερους του ενός κόμβους σε έναν, κατά τον ίδιο τρόπο που περιγράφηκε παραπάνω.

Για παράδειγμα, έστω ότι ο χρήστης επιθυμεί να προσθέσει στη σχέση TESTB ένα γνώρισμα, το οποίο να επιλέγεται και από μια σχέση VIEW. Προσθέτοντας τους απαραίτητους κόμβους και τις ακμές από την όψη στη σχέση και το γνώρισμα ATTR της όψης, ο γράφος γίνεται όπως φαίνεται στην Εικόνα 6.27.



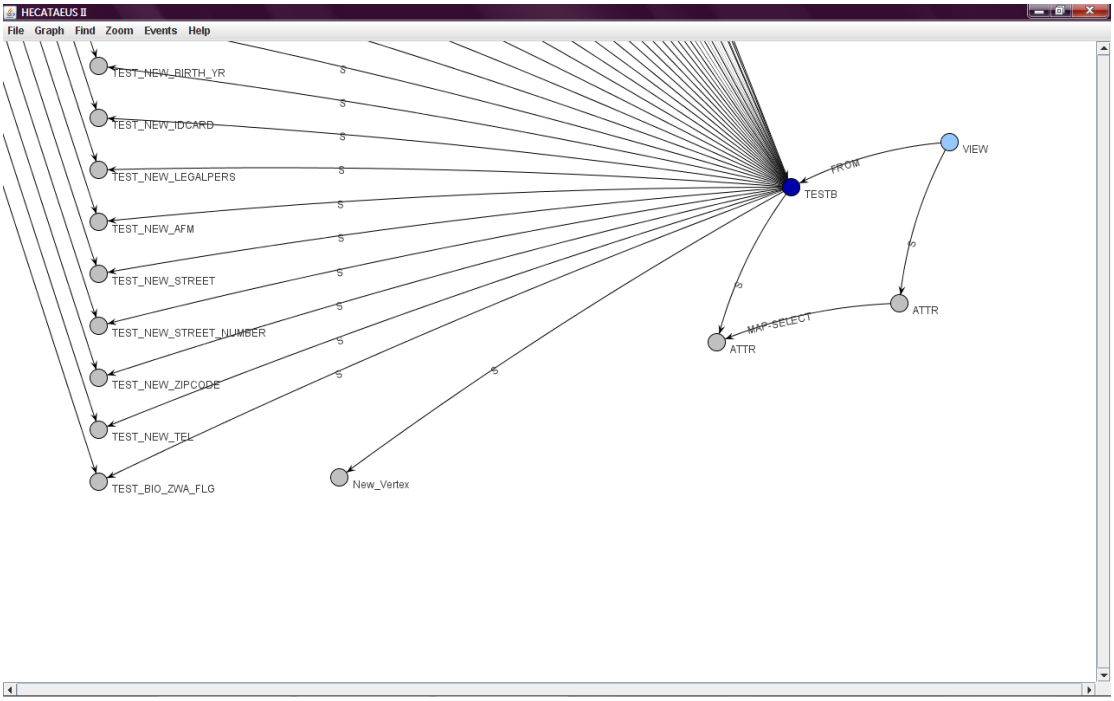
Εικόνα 6.27. Στιγμιότυπο από την προσθήκη του νέου γνωρίσματος στη σχέση TESTB και την όψη VIEW.

Τώρα πρέπει να προστεθούν οι ακμές από τη σχέση και το γνώρισμα ATTR της όψης στο νέο γνώρισμα. Επιλέγοντας τη σχέση και το γνώρισμα ATTR της όψης και με δεξί πάτημα του ποντικιού στο νέο γνώρισμα, εμφανίζονται οι δύο επιλογές δημιουργίας ακμών.



Εικόνα 6.28. Δημιουργία ακμών από τη σχέση και το γνώρισμα ATTR της όψης στο νέο γνώρισμα.

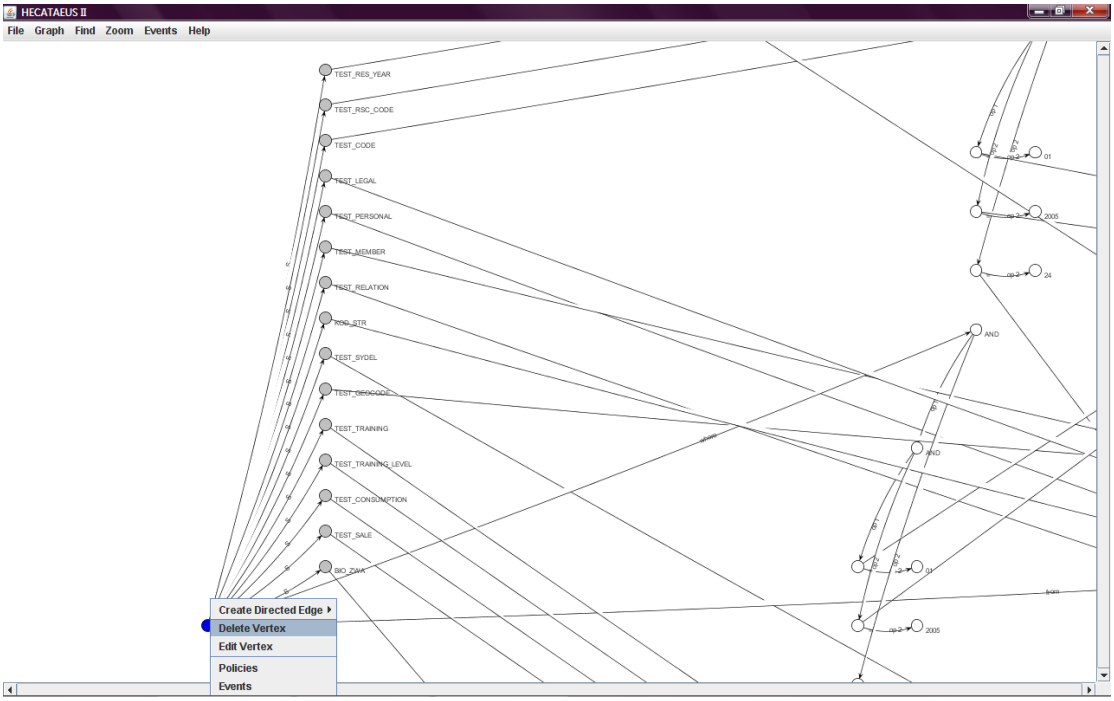
Έπειτα από την εισαγωγή των ακμών ο γράφος γίνεται όπως φαίνεται στην Εικόνα 6.29.



Εικόνα 6.29. Ο γράφος μετά τη δημιουργία ακμών από τη σχέση και το γνώρισμα ATTR της όψης στο νέο γνώρισμα.

6.2.2.2 Διαγραφή κόμβων-Διαγραφή ακμών

Από τις επιλογές που εμφανίζονται για έναν κόμβο με δεξί πάτημα του ποντικιού μπορεί ο χρήστης να διαγράψει το συγκεκριμένο κόμβο. Έστω ότι ο χρήστης θέλει να διαγράψει το ερώτημα Q2. Με δεξί πάτημα του ποντικιού και επιλέγοντας «Delete Vertex», εμφανίζεται ένα παράθυρο προειδοποίησης, στο οποίο ζητείται ο χρήστης να επιβεβαιώσει την επιθυμία του για διαγραφή του κόμβου.



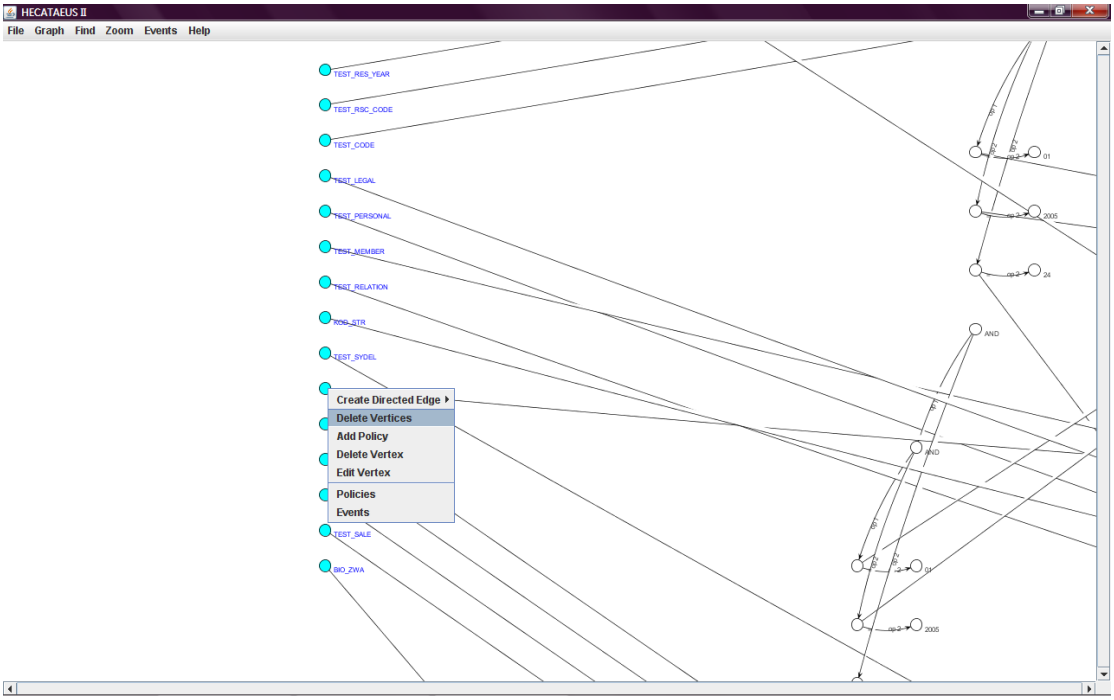
Εικόνα 6.30. Διαγραφή του ερωτήματος Q2.

Μετά την επιβεβαίωση, σβήνεται από το πλάνο ο επιλεγμένος κόμβος, καθώς και οι ακμές που συνδέονται με αυτόν.

Ο χρήστης μπορεί να διαγράψει ταυτόχρονα και περισσότερους του ενός κόμβους, επιλέγοντάς τους και πατώντας «Delete Vertices» στο μενού που εμφανίζεται με δεξί πάτημα του ποντικιού πάνω σε έναν από αυτούς. Για παράδειγμα, η διαγραφή των κόμβων-παιδιών του διαγραμμένου ερωτήματος Q2 μπορούν να διαγραφούν όπως φαίνεται στην Εικόνα 6.31.

Οι επιλεγμένοι κόμβοι διαγράφονται μετά την επιβεβαίωση από το χρήστη σε αντίστοιχο προειδοποιητικό μήνυμα.

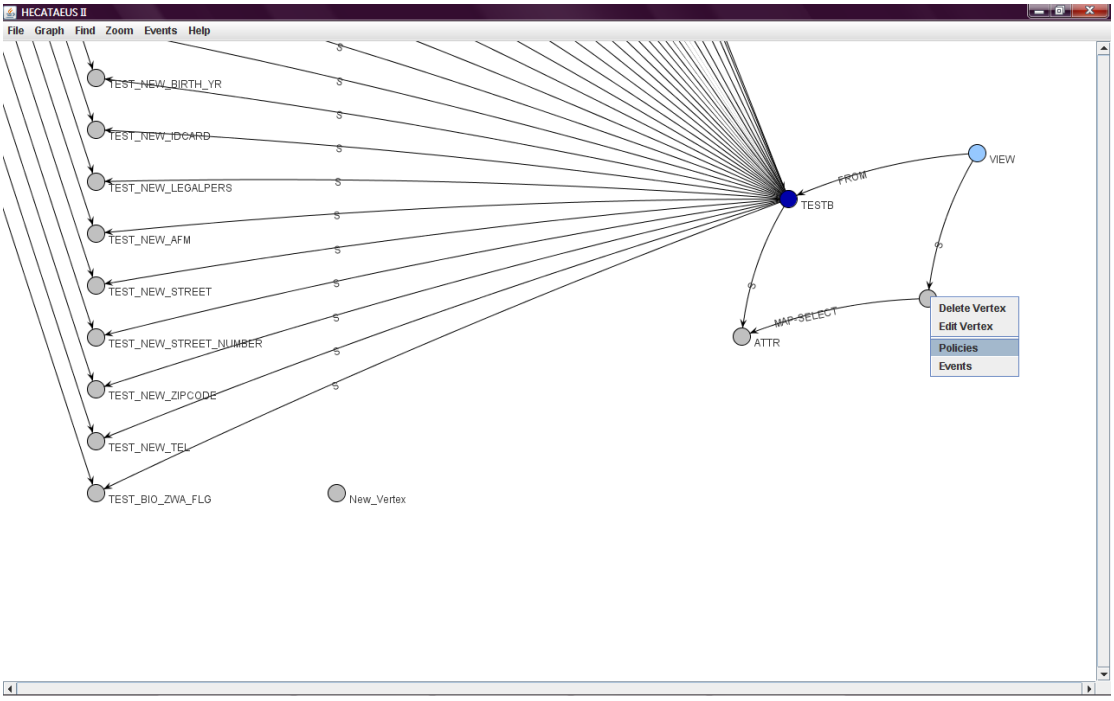
Μια ακμή διαγράφεται κατά τον ίδιο τρόπο. Για παράδειγμα, με δεξί πάτημα του ποντικιού πάνω στην ακμή και επιλέγοντας «Delete Edge» μπορεί ο χρήστης να διαγράψει την ακμή από τη σχέση TESTB στο γνώρισμα New_Vertex, αφού επιβεβαιώσει την επιθυμία του στο σχετικό παράθυρο προειδοποίησης.



Εικόνα 6.31. Διαγραφή περισσότερων του ενός κόμβων.

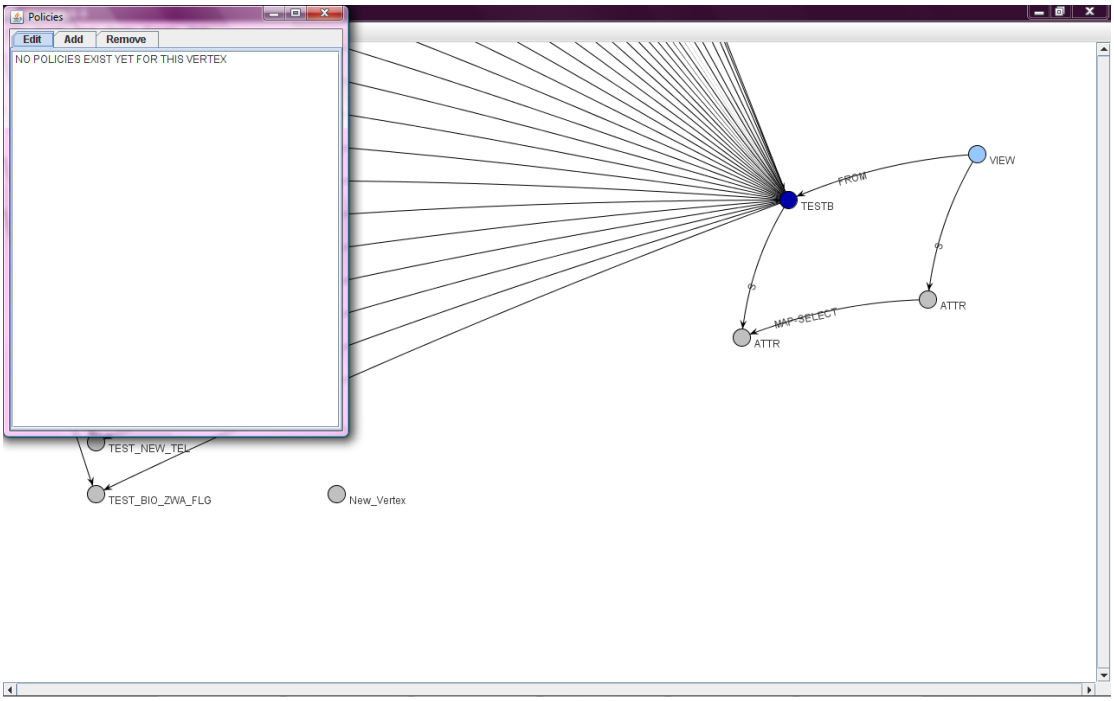
6.2.3 Επιλογές για πολιτικές

Με δεξί πάτημα πάνω σε έναν κόμβο, και επιλέγοντας «Policies», εμφανίζεται ένα παράθυρο επιλογών, στο οποίο ο χρήστης έχει διάφορες επιλογές όσον αφορά τις πολιτικές του συγκεκριμένου κόμβου. Για παράδειγμα, για τον κόμβο ATTR της όψης VIEW εμφανίζεται το παράθυρο όπως φαίνεται στην Εικόνα 6.33.



Εικόνα 6.32. Άνοιγμα του παραθύρου πολιτικών για τον κόμβο ATTR της όψης VIEW.

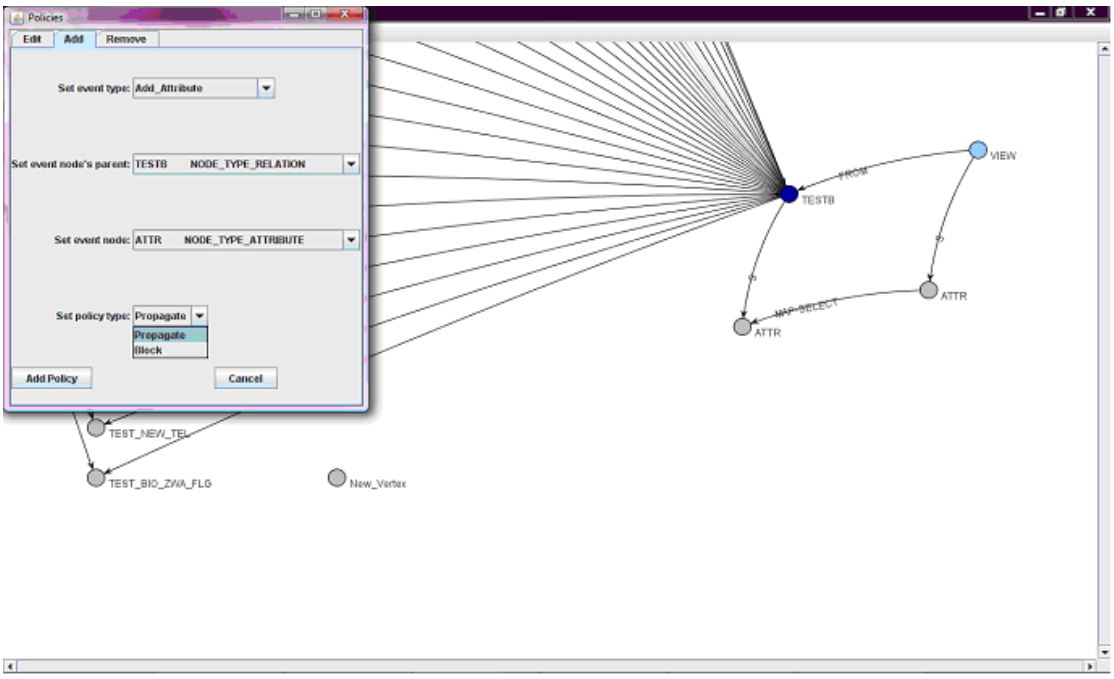
Στην καρτέλα «Edit» γράφει «NO POLICIES EXIST YET FOR THIS VERTEX», επειδή ο κόμβος δεν έχει ακόμα καμία πολιτική.



Εικόνα 6.33. Παράθυρο πολιτικών.

6.2.3.1 Προσθήκη πολιτικής

Επιλέγοντας από το παράθυρο των πολιτικών «Add» εμφανίζεται η καρτέλα για την προσθήκη νέας πολιτικής στο συγκεκριμένο κόμβο, όπως φαίνεται στην Εικόνα 6.34.

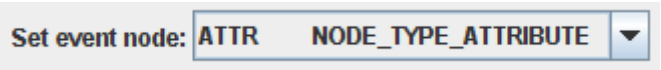


Εικόνα 6.34. Προσθήκη νέας πολιτικής

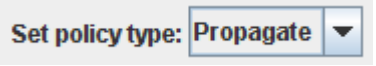
Η καρτέλα περιέχει λίστες για την επιλογή του γεγονότος που χειρίζεται η πολιτική:

την επιλογή του κόμβου-πατέρα του κόμβου τον οποίο αφορά το γεγονός (ο κόμβος-πατέρας είναι τύπου «σχέση», «ερώτημα» ή «όψη» αν ο κόμβος που αφορά το γεγονός είναι τύπου «γνώρισμα», «περιορισμός» ή «σταθερά», ή «null» αλλιώς):

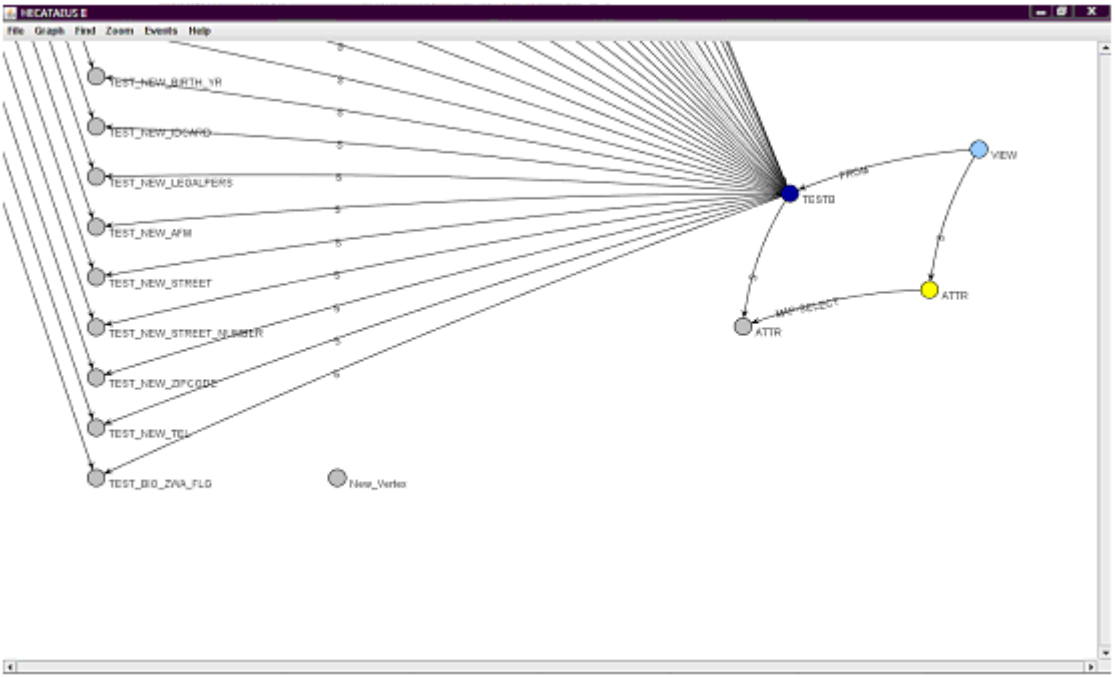
την επιλογή του κόμβου τον οποίο αφορά το γεγονός (ανάλογα με τον κόμβο-πατέρα εμφανίζονται οι προς επιλογή κόμβοι τύπου «γνώρισμα», «περιορισμός» ή «σταθερά» ή όλοι οι κόμβοι διαφορετικού τύπου αν ο κόμβος-πατέρας είναι «null»):



και την επιλογή του τύπου της πολιτικής:



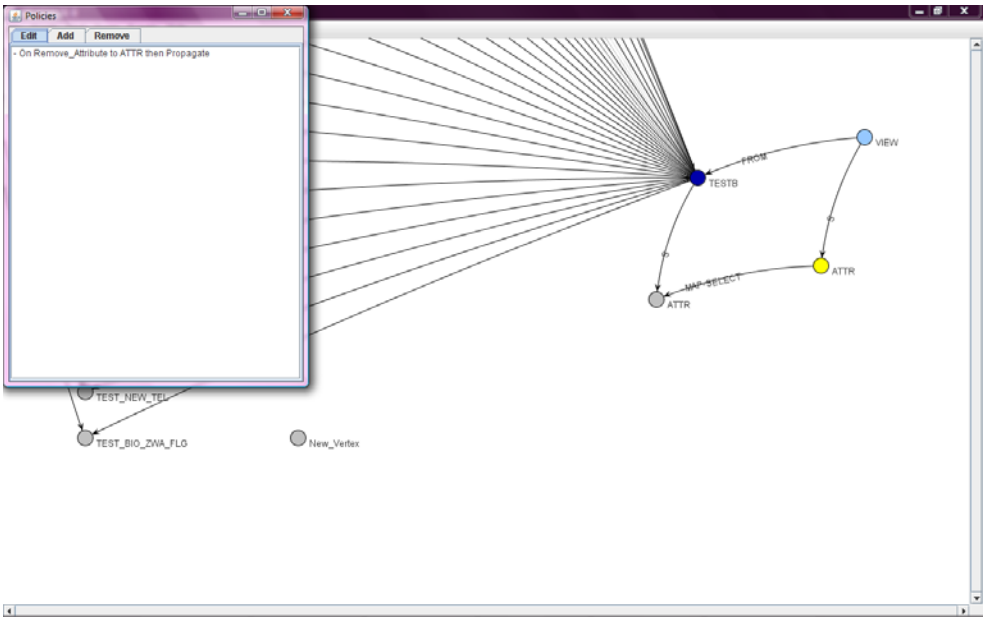
Επιλέγοντας ως τύπο γεγονότος «Αφαίρεση γνωρίσματος», ως κόμβο γεγονότος το γνώρισμα ATTR της σχέσης TESTB και ως τύπο πολιτικής «Προώθηση» και πατώντας το κουμπί «Add Policy» εξαφανίζεται το παράθυρο των πολιτικών και ο κόμβος στον οποίο ορίστηκε η πολιτική χρωματίζεται κίτρινος, δηλώνοντας ότι έχει πολιτική.



Εικόνα 6.35. Ο κόμβος γνωρίσματος ATTR έχει μια πολιτική.

6.2.3.2 Εμφάνιση ορισμένων πολιτικών

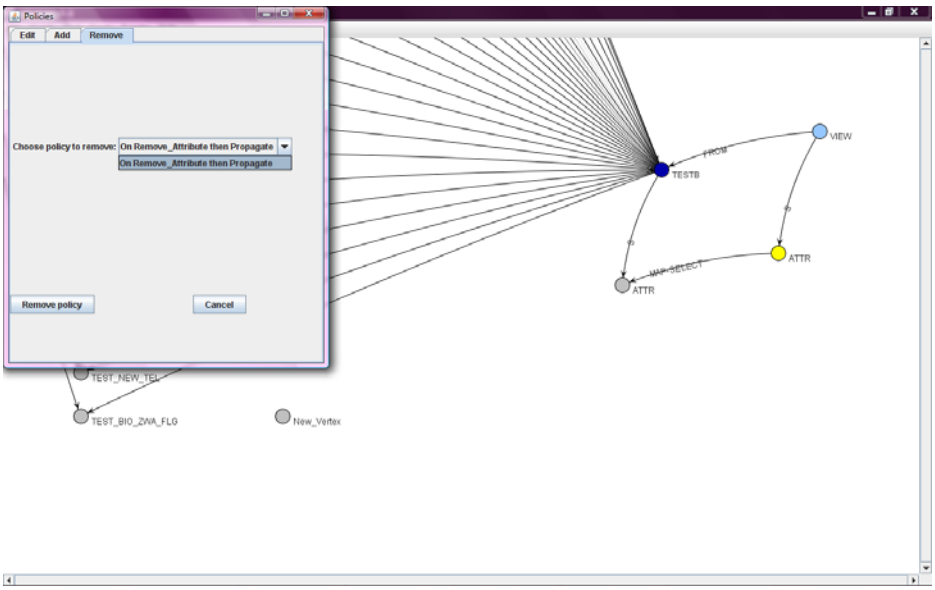
Επιλέγοντας από το παράθυρο των πολιτικών «Edit» εμφανίζεται η καρτέλα για την εμφάνιση όλων των πολιτικών που έχουν οριστεί για το συγκεκριμένο κόμβο. Ο κόμβος έχει μόνο μια πολιτική, η οποία και εμφανίζεται.



Εικόνα 6.36. Εμφάνιση των πολιτικών του κόμβου ATTR.

6.2.3.3 Αφαίρεση ορισμένης πολιτικής

Επιλέγοντας από το παράθυρο των πολιτικών «Remove» εμφανίζεται η καρτέλα για την αφαίρεση πολιτικής από το συγκεκριμένο κόμβο. Η καρτέλα έχει μια λίστα για την επιλογή μιας από της πολιτικές του κόμβου προς αφαίρεση. Για τον συγκεκριμένο κόμβο υπάρχει μια μόνο πολιτική.

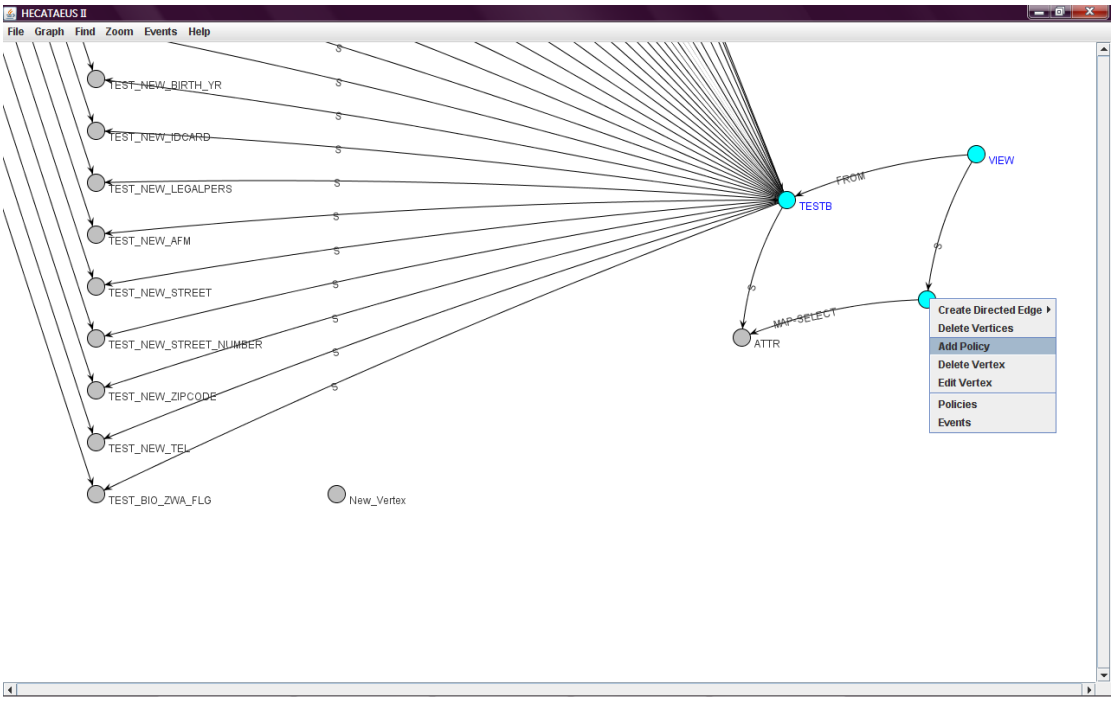


Εικόνα 6.37. Αφαίρεση πολιτικής από τον κόμβο ATTR.

Πατώντας το κουμπί «Remove policy» η επιλεγμένη πολιτική αφαιρείται από το σύνολο πολιτικών του κόμβου. Το χρώμα του κόμβου γίνεται πάλι γκρι, δηλώνοντας ότι δεν έχει πολιτική.

6.2.3.4 Προσθήκη πολιτικής σε πολλούς κόμβους

Ο χρήστης έχει τη δυνατότητα να ορίσει μια πολιτική σε πολλούς κόμβους ταυτόχρονα, επιλέγοντας τους κόμβους και επιλέγοντας «Add Policy» από το μενού που εμφανίζεται με δεξί πάτημα του ποντικιού πάνω σε έναν από αυτούς τους κόμβους.



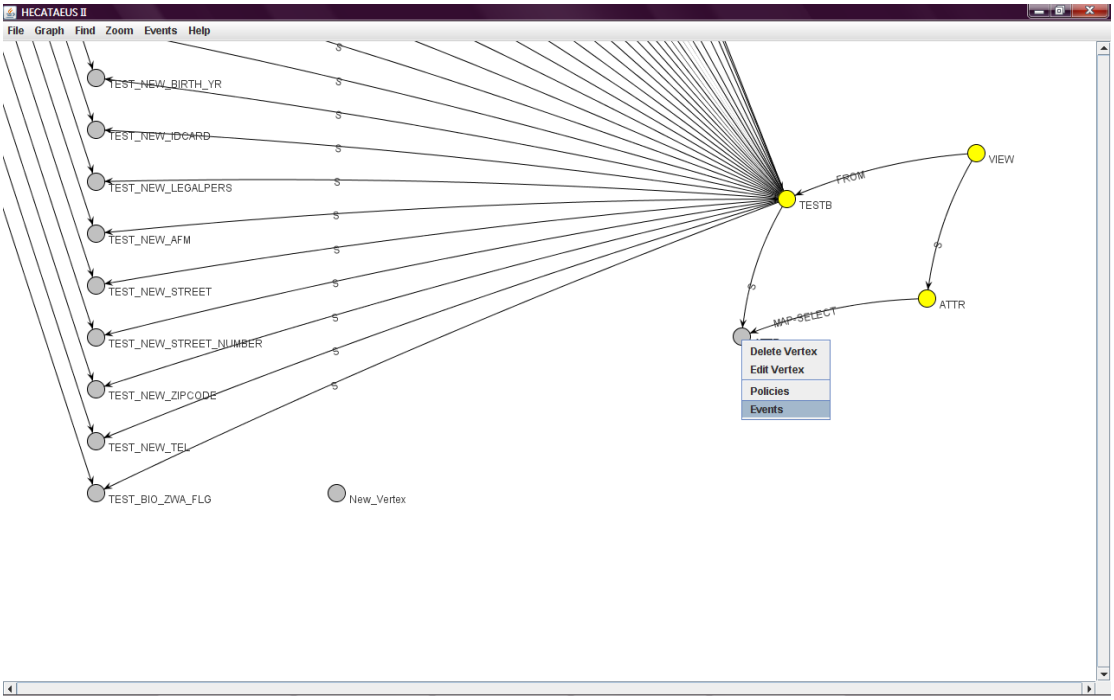
Εικόνα 6.38. Εισαγωγή πολιτικής ταυτόχρονα σε πολλούς κόμβους.

Ο ορισμός πολιτικής γίνεται στη συνέχεια κατά τον ίδιο τρόπο.

Η αφαίρεση των πολιτικών πρέπει να γίνει σε κάθε κόμβο χωριστά.

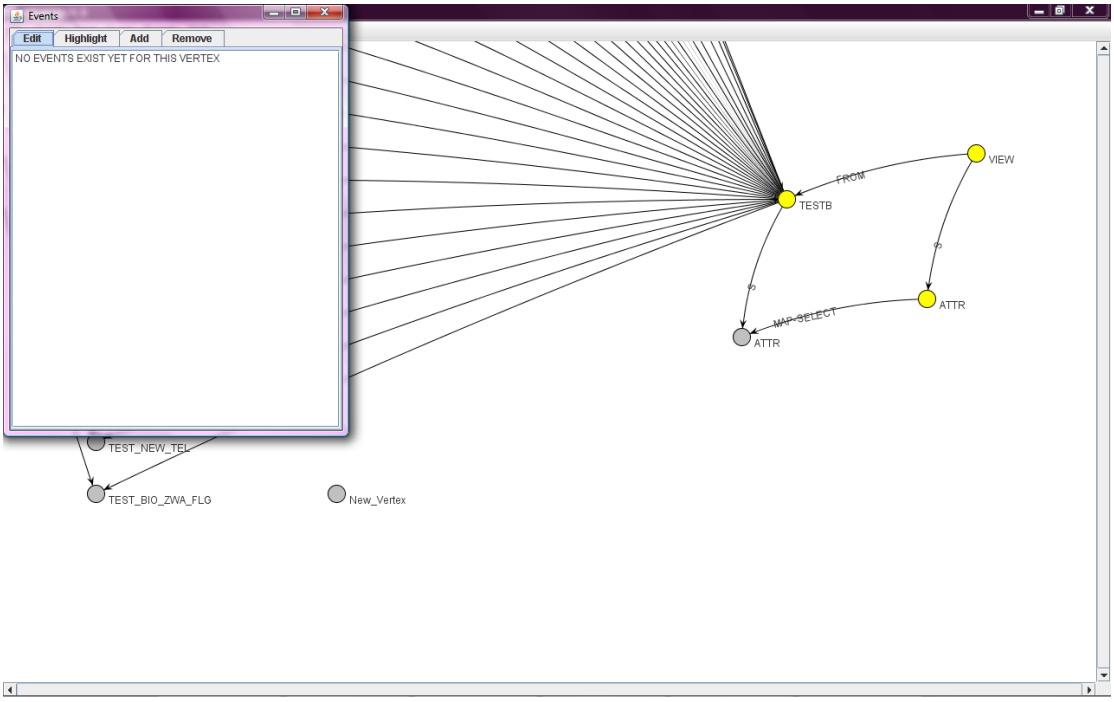
6.2.4 Επιλογές για γεγονότα

Με δεξί πάτημα πάνω σε έναν κόμβο, και επιλέγοντας «Events» (Εικόνα 6.39), εμφανίζεται ένα παράθυρο επιλογών, στο οποίο ο χρήστης έχει διάφορες επιλογές για τα γεγονότα του συγκεκριμένου κόμβου. Για παράδειγμα, για τον κόμβο ATTR της σχέσης TESTB εμφανίζεται το παράθυρο όπως φαίνεται στην Εικόνα 6.40.



Εικόνα 6.39. Άνοιγμα του παραθύρου γεγονότων για τον κόμβο ATTR της σχέσης TESTB.

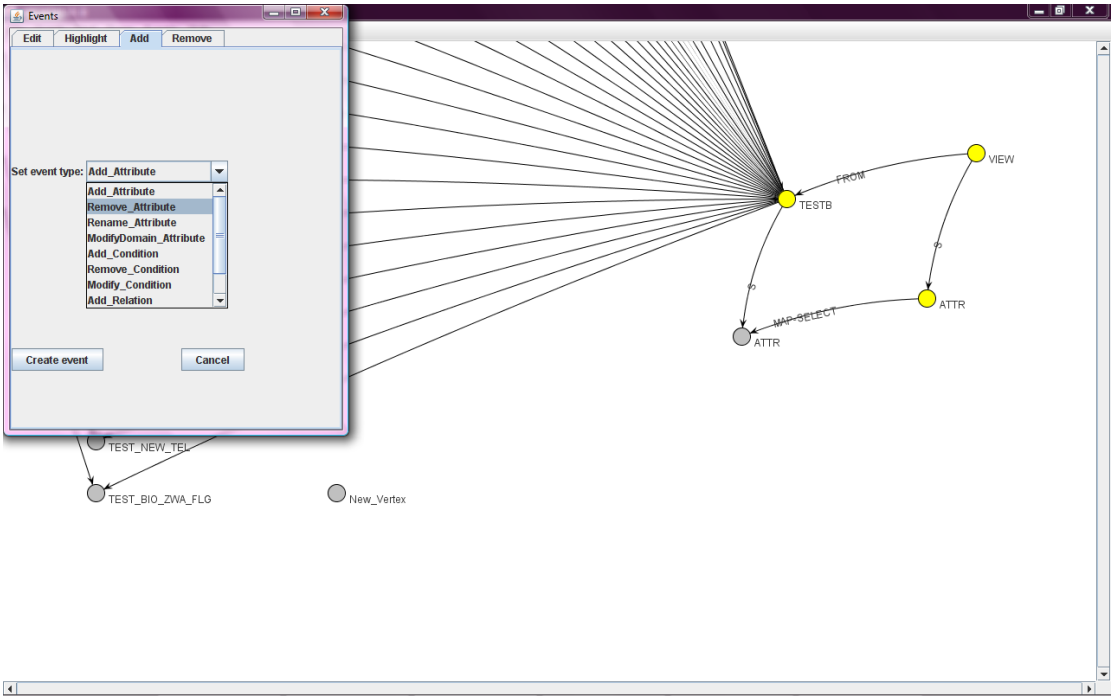
Στην καρτέλα «Edit» γράφει «NO EVENTS EXIST YET FOR THIS VERTEX», επειδή ο κόμβος δεν έχει ακόμα κανένα γεγονός.



Εικόνα 6.40. Παράθυρο γεγονότων.

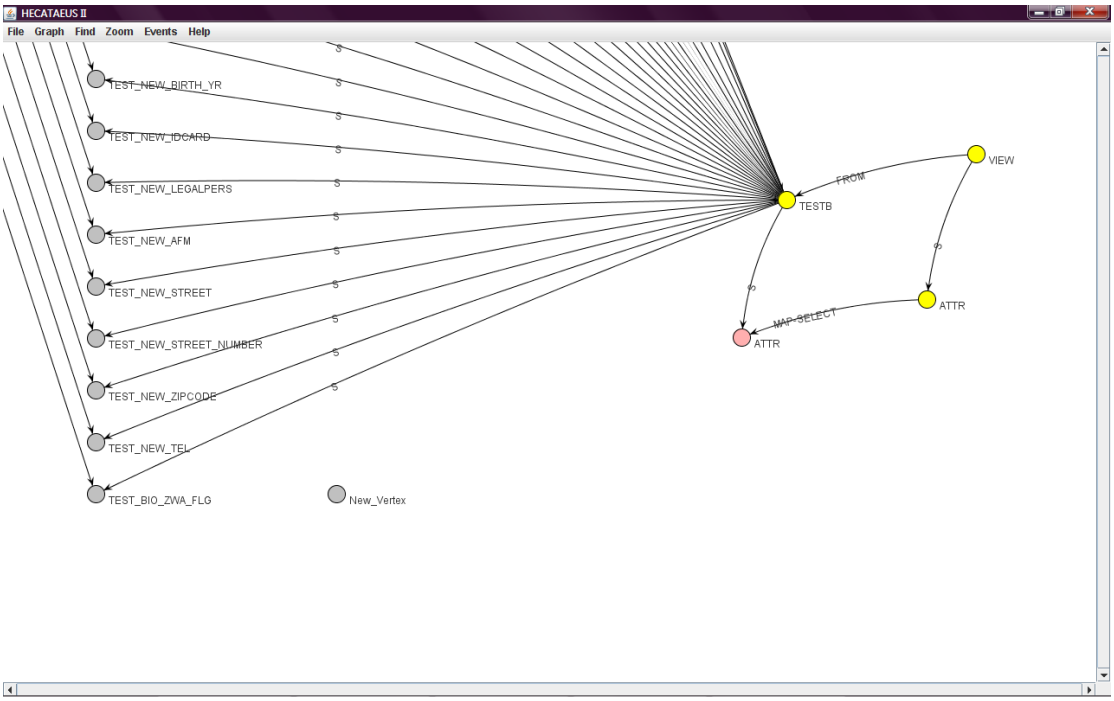
6.2.4.1 Προσθήκη γεγονότος

Επιλέγοντας από το παράθυρο των γεγονότων «Add» εμφανίζεται η καρτέλα για την προσθήκη νέου γεγονότος στο συγκεκριμένο κόμβο.



Εικόνα 6.41. Προσθήκη νέου γεγονότος

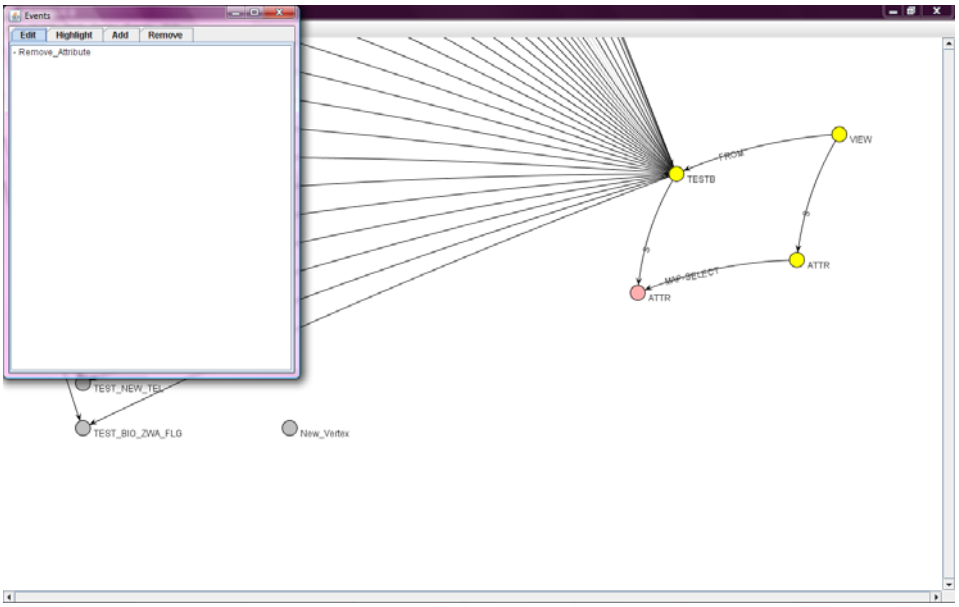
Επιλέγοντας ως τύπο γεγονότος «Αφαίρεση γνωρίσματος» και πατώντας το κουμπί «Create Event» εξαφανίζεται το παράθυρο των γεγονότων και ο κόμβος στον οποίο ορίστηκε το γεγονός χρωματίζεται ροζ, δηλώνοντας ότι έχει γεγονός.



Εικόνα 6.42. Ο κόμβος γνωρίσματος ATTR έχει ένα γεγονός.

6.2.4.2 Εμφάνιση ορισμένων γεγονότων

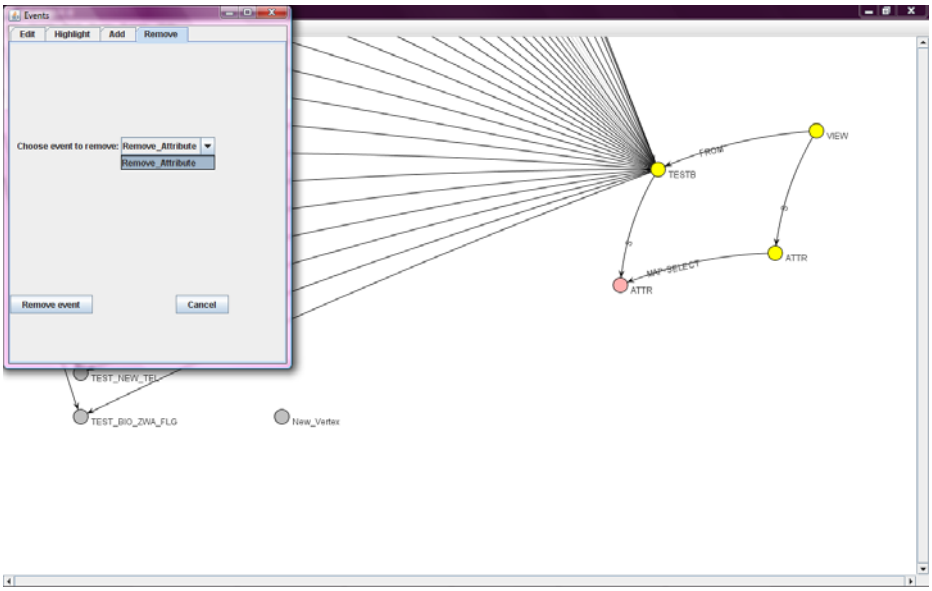
Επιλέγοντας από το παράθυρο των γεγονότων «Edit» εμφανίζεται η καρτέλα για την εμφάνιση όλων των γεγονότων που έχουν οριστεί για το συγκεκριμένο κόμβο. Ο κόμβος έχει μόνο ένα γεγονός, το οποίο και εμφανίζεται.



Εικόνα 6.43. Εμφάνιση των γεγονότων του κόμβου ATTR.

6.2.4.3 Αφαίρεση ορισμένου γεγονότος

Επιλέγοντας από το παράθυρο των γεγονότων «Remove» εμφανίζεται η καρτέλα για την αφαίρεση γεγονότος από το συγκεκριμένο κόμβο. Η καρτέλα έχει μια λίστα για την επιλογή ενός από τα γεγονότα του κόμβου προς αφαίρεση. Για τον συγκεκριμένο κόμβο υπάρχει ένα μόνον γεγονός.



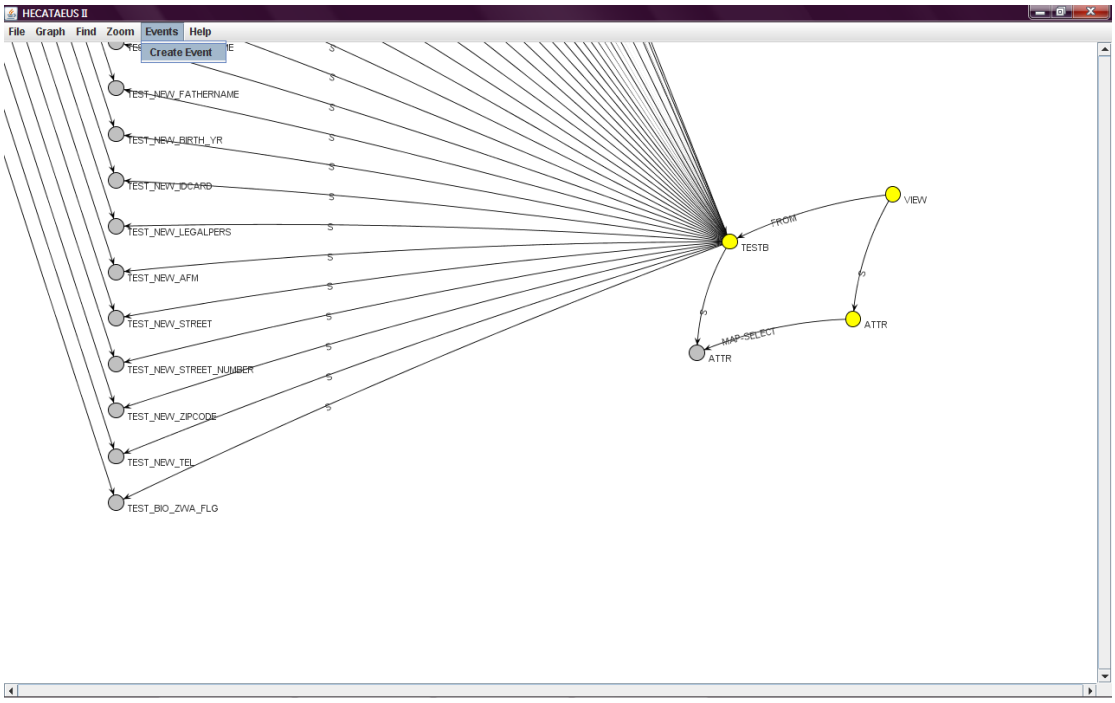
Εικόνα 6.44. Αφαίρεση γεγονότος από τον κόμβο ATTR.

Πατώντας το κουμπί «Remove event» το επιλεγμένο γεγονός αφαιρείται από το σύνολο γεγονότων του κόμβου. Το χρώμα του κόμβου γίνεται πάλι γκρι, δηλώνοντας ότι δεν έχει γεγονός.

6.2.4.4 Εναλλακτικός ορισμός γεγονότος από το μενού επιλογών.

Ο χρήστης μπορεί να ορίσει ένα γεγονός επιλέγοντας τον κόμβο που αφορά το γεγονός και τον τύπο του γεγονότος από ένα παράθυρο επιλογών. Αυτό είναι χρήσιμο όταν στο πλάνο υπάρχουν πολλοί κόμβοι, ούτως ώστε ο χρήστης να μη χρειάζεται να βρίσκει τον κόμβο που αφορά το γεγονός στο πλάνο.

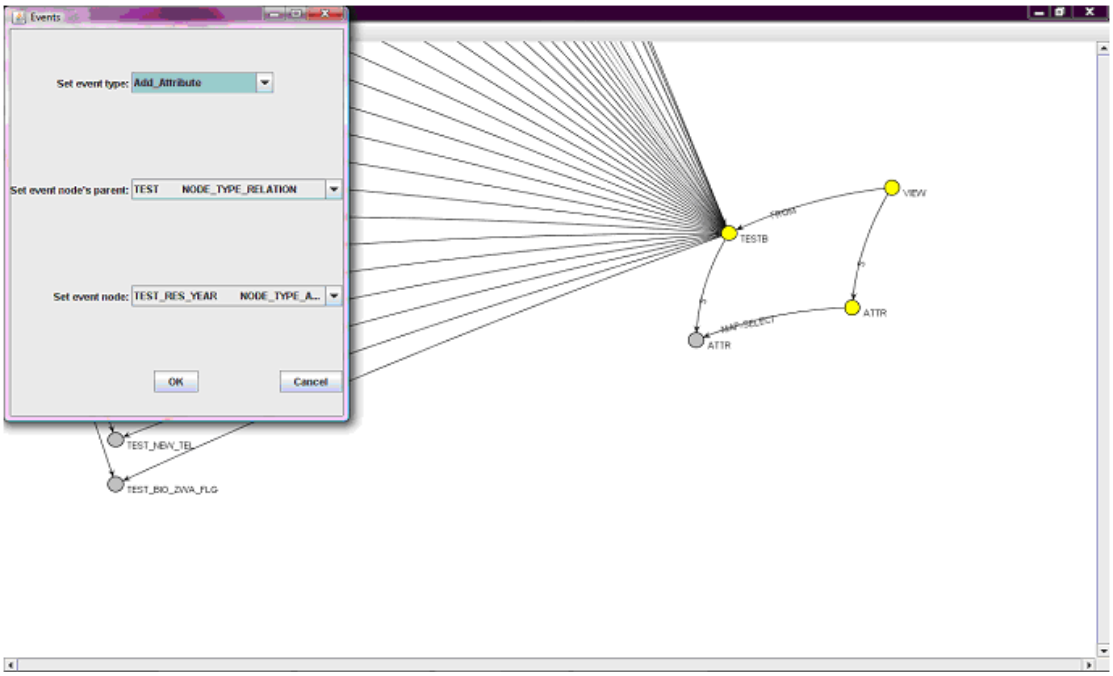
Έστω ότι ο χρήστης επιθυμεί να ορίσει ένα γεγονός τύπου «διαγραφή γνωρίσματος» στον κόμβο ATTR της σχέσης TESTB. Επιλέγει «Create Event» από το «Events» του μενού επιλογών, όπως φαίνεται στην Εικόνα 6.45.



Εικόνα 6.45. Ορισμός νέου γεγονότος από το μενού επιλογών.

Στο παράθυρο που εμφανίζεται (Εικόνα 6.46) ο χρήστης επιλέγει τον τύπο του γεγονότος, τον κόμβο-πατέρα του κόμβου που αφορά το γεγονός και τον κόμβο που αφορά το γεγονός. Στην προκειμένη περίπτωση ο τύπος του γεγονότος είναι «διαγραφή γνωρίσματος», ο κόμβος-πατέρας είναι η σχέση TESTB και ο κόμβος που αφορά το γεγονός το γνώρισμα ATTR.

Κάνοντας τις κατάλληλες επιλογές και πατώντας «OK» το γεγονός προστίθεται στο γνώρισμα ATTR και το χρώμα του γίνεται ροζ.



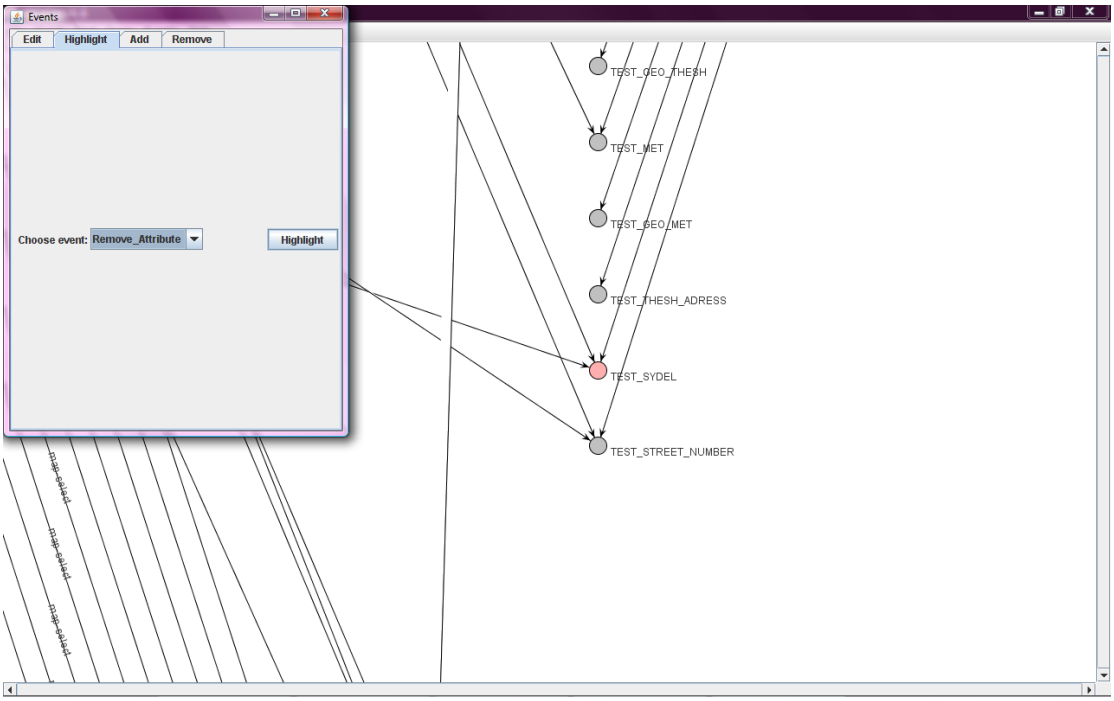
Εικόνα 6.46. Ορισμός νέου γεγονότος.

6.2.5 Σενάρια εξέλιξης βάσεων δεδομένων.

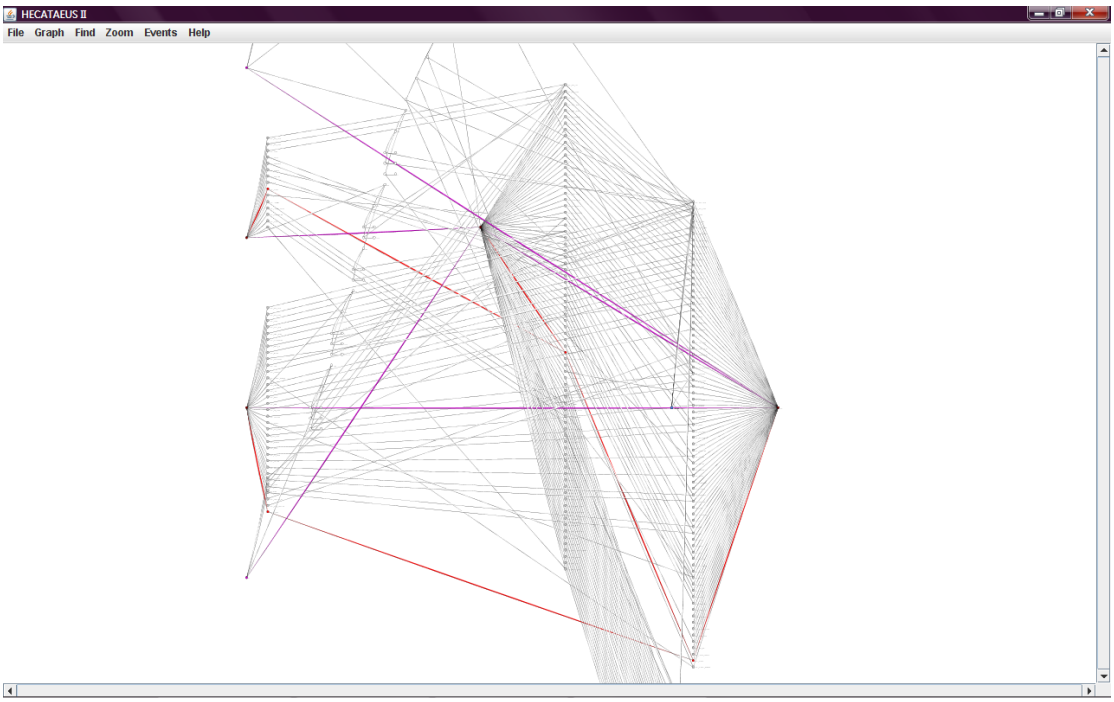
Ο ΕΚΑΤΑΙΟΣ ΙΙ προσφέρει τη δυνατότητα εφαρμογής πιθανών σεναρίων εξέλιξης του σχήματος της βάσης δεδομένων και εύρεσης του τρόπου με τον οποίο τα διάφορα τμήματα του συστήματος επηρεάζονται από αυτή την αλλαγή. Έστω ότι ο χρήστης επιθυμεί να διαγράψει το γνώρισμα TEST_SYDEL της σχέσης TEST. Ορίζοντας ένα νέο γεγονός «διαγραφή γνωρίσματος», επιλέγοντας από την καρτέλα «Highlight» του παραθύρου των γεγονότων το γεγονός «διαγραφή γνωρίσματος» του γνωρίσματος TEST_SYDEL της σχέσης TEST και πατώντας το κουμπί «Highlight» (Εικόνα 6.47), τα στοιχεία του γράφου που επηρεάζονται από αυτό το γεγονός αλλάζουν κατάσταση και συνεπώς χρώμα, όπως φαίνεται στην Εικόνα 6.48.

Επειδή η προεπιλεγμένη πολιτική των κόμβων είναι προώθηση, ο κόμβος TEST_SYDEL της σχέσης TEST γίνεται κόκκινος, δηλαδή «προς διαγραφή». Όλες οι ακμές που προσπίπτουν στον κόμβο αυτό γίνονται επίσης κόκκινες («προς διαγραφή»). Τα γνωρίσματα TEST_SYDEL των ερωτημάτων Q2 και Q3 και της όψης TEST_V που επιλέγουν από το γνώρισμα TEST_SYDEL της σχέσης TEST αλλάζουν την κατάστασή τους, η οποία γίνεται «προς διαγραφή» (κόκκινο χρώμα). Η σχέση TEST, η όψη TEST_V και τα ερωτήματα Q2 και Q3 αλλάζουν την κατάστασή τους σε «ένας κόμβος-παιδί μου διαγράφεται» και το χρώμα τους σε σκούρο κόκκινο. Οι κόμβοι με μοβ χρώμα δηλώνουν κόμβους των οποίων η κατάσταση είναι «επηρασμένος» και αφορούν ερωτήματα ή όψεις που προσπελάζουν κόμβους (σχέσεις ή όψεις, αντίστοιχα) που επηρεάζονται με οποιοδήποτε τρόπο. Οι ακμές με

μοβ χρώμα δηλώνουν ακμές που δεν είναι προς διαγραφή, αλλά καταλήγουν σε κόμβους οι οποίοι επηρεάζονται.

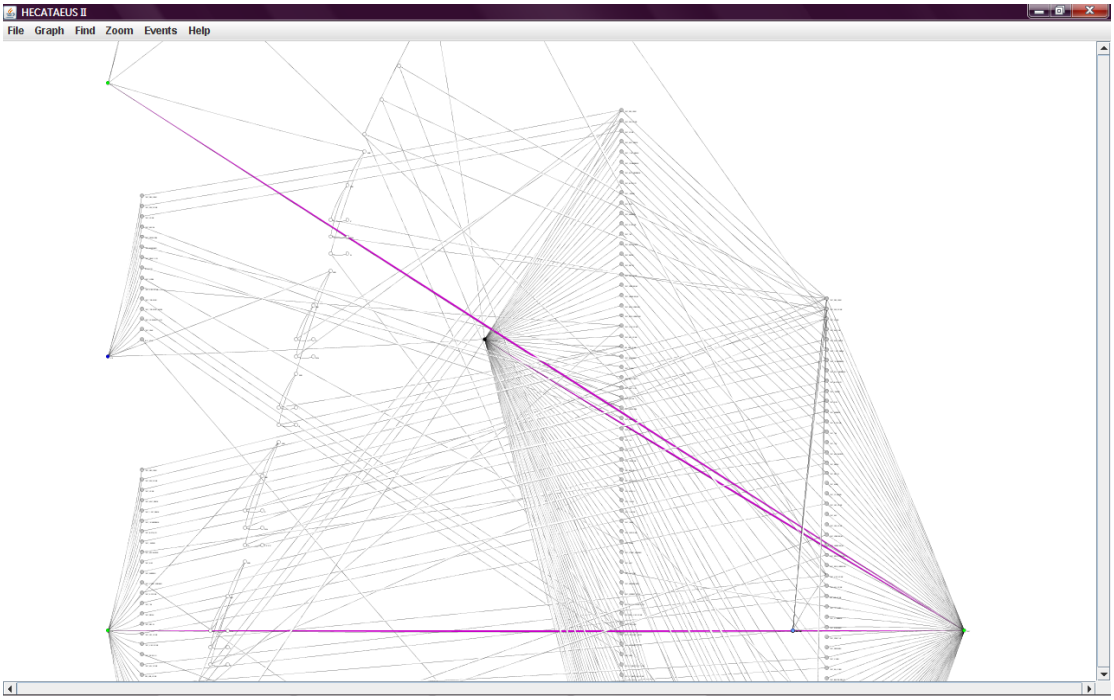


Εικόνα 6.47. Επιλογή για τονισμό των στοιχείων του γράφου που επηρεάζονται από το γεγονός «διαγραφή γνωρίσματος» του κόμβου TEST_SYDEL.



Εικόνα 6.48. Τονισμός των στοιχείων του γράφου που επηρεάζονται από το γεγονός «διαγραφή γνωρίσματος» του κόμβου TEST_SYDEL.

Έστω τώρα το σενάριο της εισαγωγής ενός γνωρίσματος στη σχέση TEST. Ορίζοντας ένα νέο γεγονός «προσθήκη γνωρίσματος» στη σχέση TEST, μια πολιτική «παρεμπόδιση» για το συγκεκριμένο γεγονός στην όψη TEST_V και εφαρμόζοντας το γεγονός, όπως περιγράφηκε παραπάνω, τα στοιχεία του γράφου που επηρεάζονται από αυτό το γεγονός αλλάζουν κατάσταση και χρώμα (Εικόνα 6.49).



Εικόνα 6.49. Τονισμός των στοιχείων του γράφου που επηρεάζονται από το γεγονός «διαγραφή γνωρίσματος» του κόμβου TEST_SYDEL.

Η σχέση TEST αλλάζει την κατάστασή της σε «προς προσθήκη κόμβου-παιδιού» και το χρώμα της σε πράσινο. Ομοίως τα ερωτήματα Q1 και Q3 που προσπελαύνουν άμεσα τη σχέση (δηλαδή όχι μέσω της όψης) γίνονται πράσινα, δηλώνοντας ότι είναι πιθανό να χρειάζεται σ' αυτά η προσθήκη ενός γνωρίσματος που θα επιλέγει από το νέο γνώρισμα που προστίθεται στη σχέση. Αντίθετα, η όψη TEST_V γίνεται μαύρη, δηλώνοντας ότι εμποδίζει την αλλαγή όπως υπαγορεύεται από την πολιτική της, δεν επηρεάζεται από αυτή και δεν την προωθεί στα ερωτήματα που την προσπελαύνουν (Q2).

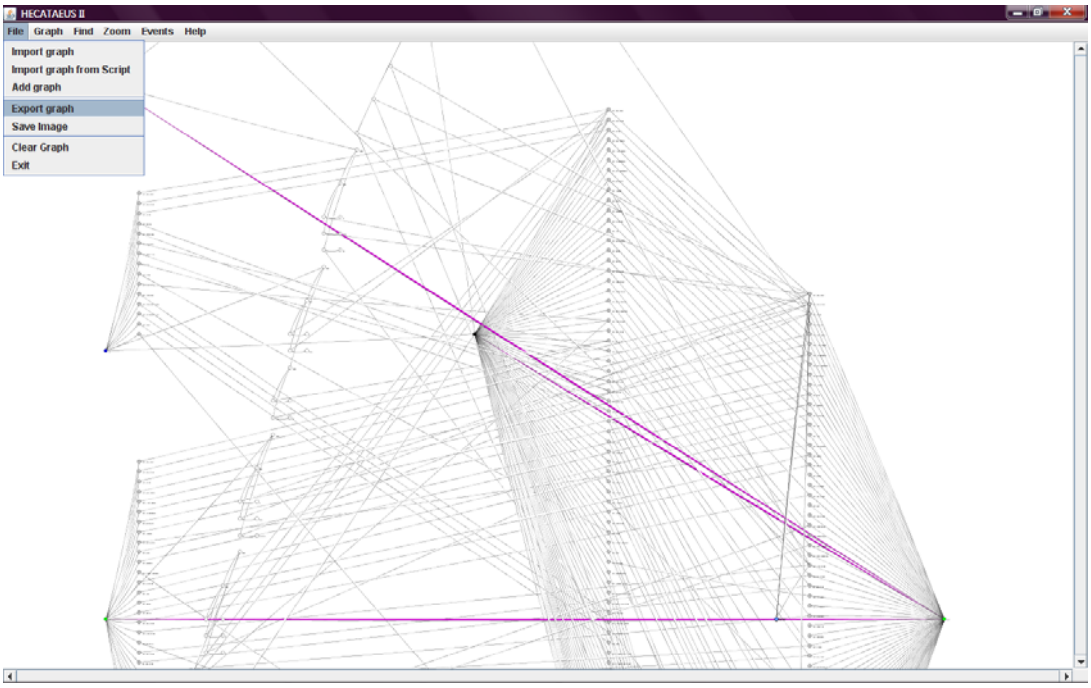
6.2.6 Αποθήκευση γράφου.

Έπειτα από τις αλλαγές που έγιναν στο γράφο, ο χρήστης ίσως επιθυμεί να αποθηκεύσει την τελική κατάσταση. Υπάρχουν δύο δυνατότητες για αποθήκευση του γράφου:

- Σε αρχείο xml.
- Ως εικόνα.

6.2.6.1 Αποθήκευση γράφου σε αρχείο xml.

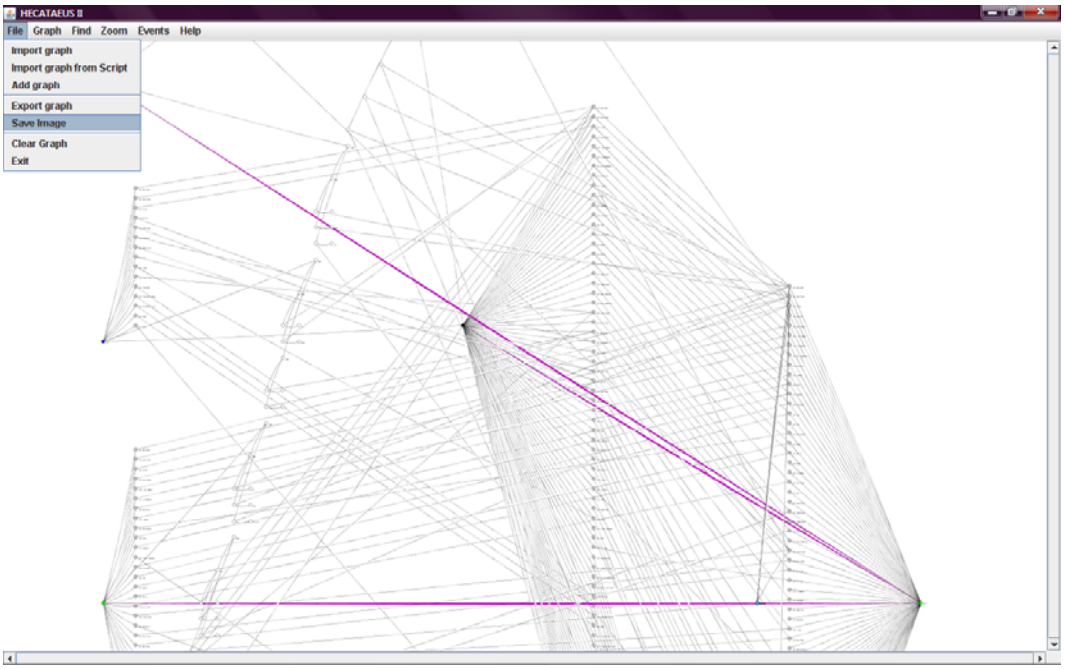
Επιλέγοντας «Export graph» από το «File» του μενού επιλογών, ο χρήστης μπορεί να αποθηκεύσει τον οπτικοποιημένο γράφο σε ένα αρχείο τύπου xml.



Εικόνα 6.50. Αποθήκευση γράφου σε αρχείο xml.

6.2.6.2 Αποθήκευση γράφου ως εικόνα.

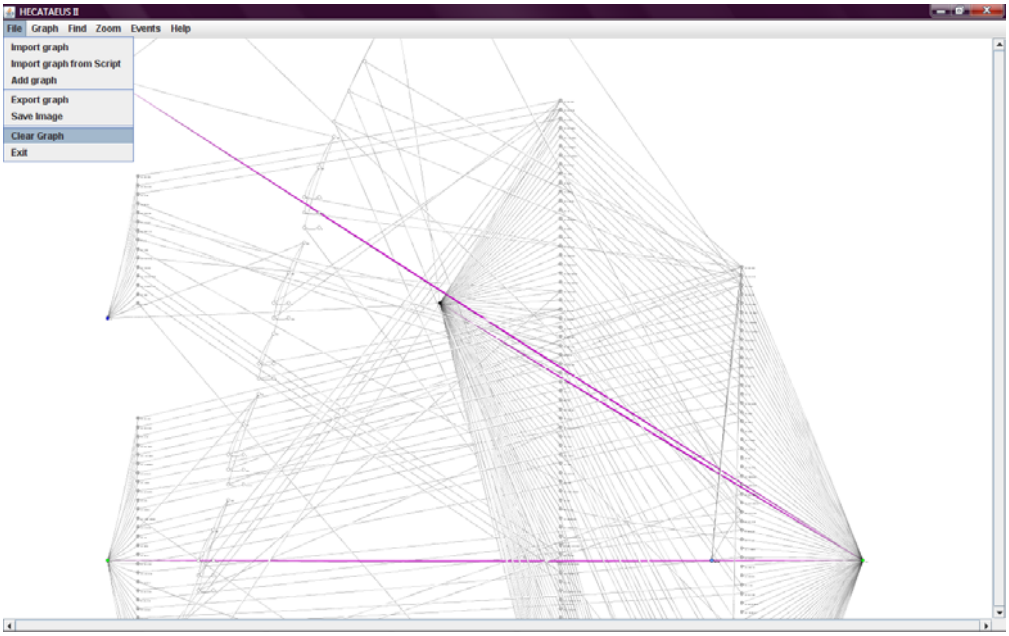
Επιλέγοντας «Save Image» από το «File» του μενού επιλογών, ο χρήστης μπορεί να αποθηκεύσει τον οπτικοποιημένο γράφο ως εικόνα.



Εικόνα 6.51. Αποθήκευση του τμήματος του γράφου που φαίνεται στο πλάνο ως εικόνα.

6.2.7 Εκκαθάριση του πλάνου-σβήσιμο των στοιχείων του γράφου-έξοδος από την εφαρμογή.

Τέλος, έστω ότι ο χρήστης επιθυμεί να εισάγει έναν καινούριο γράφο στο σύστημα. Επιλέγοντας «Clear Graph» από το «File» του μενού επιλογών ο γράφος που υπάρχει στο πλάνο σβήνεται, όπως και κάθε πληροφορία γι' αυτόν.



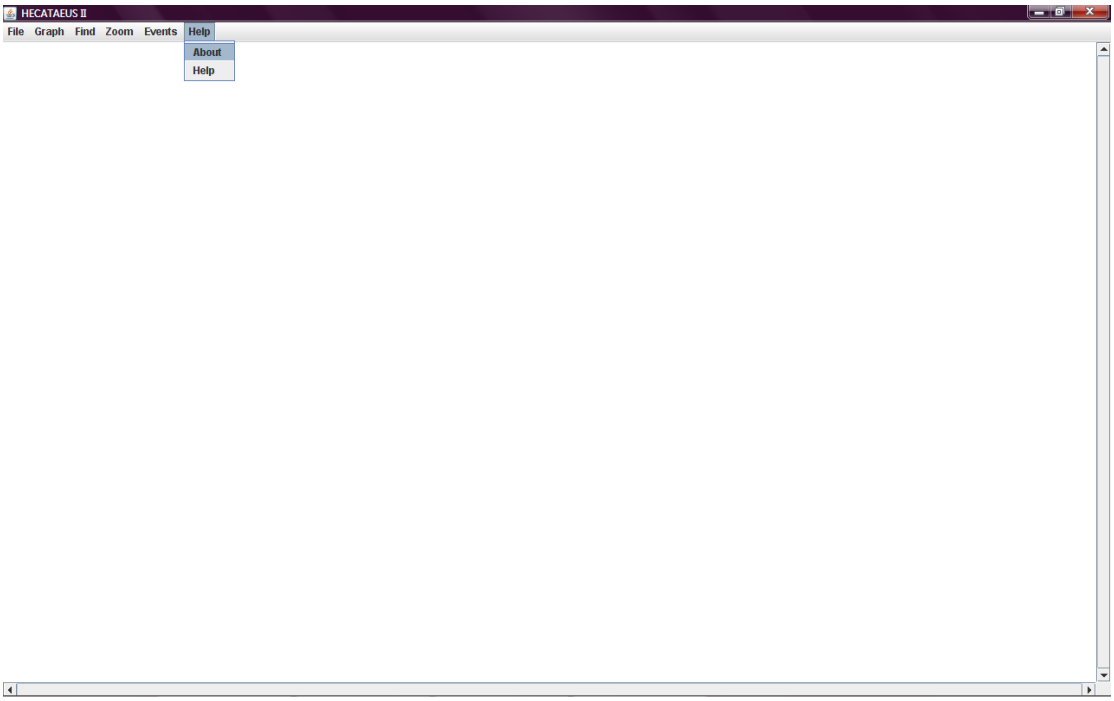
Εικόνα 6.52. Εκκαθάριση πλάνου-σβήσιμο στοιχείων γράφου.

Η έξοδος από την εφαρμογή μπορεί να γίνει απλώς κλείνοντας το βασικό παράθυρο, ή, εναλλακτικά, επιλέγοντας «Exit» από το «File» του μενού επιλογών.

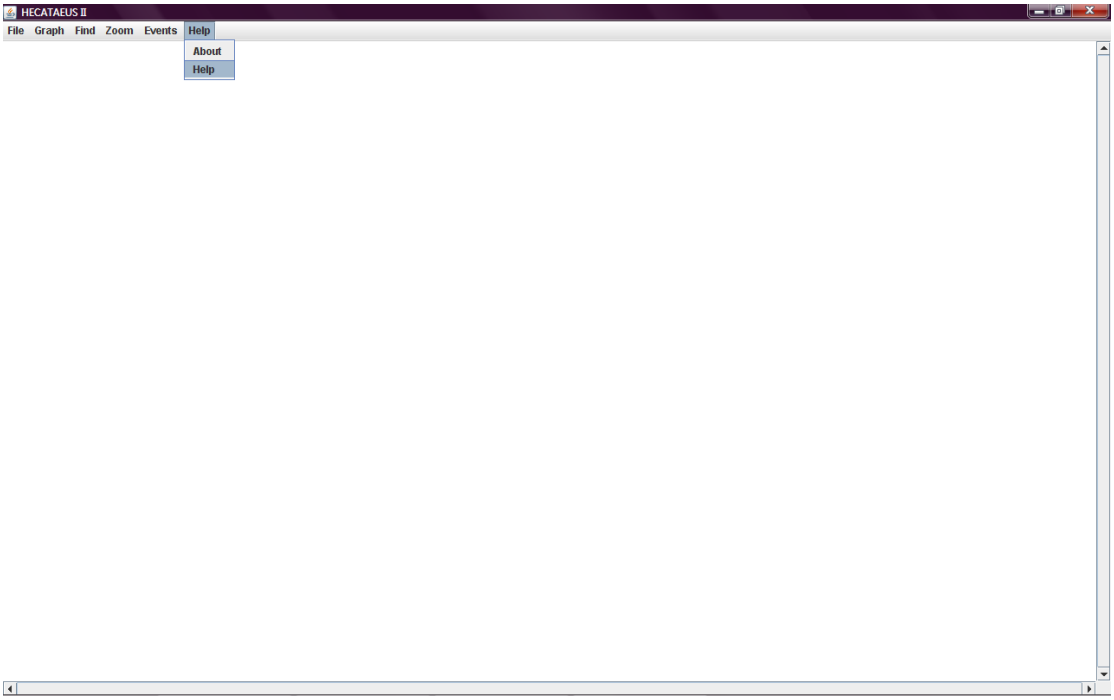
Εικόνα 6.53. Έξοδος από την εφαρμογή.

6.2.8 Σχετικά με τον ΕΚΑΤΑΙΟ ΙΙ - Βοήθεια για τον ΕΚΑΤΑΙΟ ΙΙ.

Υπάρχει, τέλος, η δυνατότητα για εμφάνιση πληροφοριών σχετικά με τον ΕΚΑΤΑΙΟ ΙΙ και βοήθεια για τις λειτουργίες του, επιλέγοντας «About» και «Help», αντίστοιχα, από το «Help» του μενού επιλογών.



Εικόνα 6.54. Εμφάνιση πληροφοριών σχετικά με τον ΕΚΑΤΑΙΟ ΙΙ.



Εικόνα 6.55. Εμφάνιση βοήθειας για τις λειτουργίες του ΕΚΑΤΑΙΟΥ ΙΙ.

7 Επίλογος

Στην ενότητα αυτή συνοψίζεται η παρουσίαση της διπλωματικής και αναφέρονται πιθανά θέματα για επέκταση της διπλωματικής.

7.1 Σύνοψη και συμπεράσματα

Σκοπός της διπλωματικής εργασίας ήταν η δημιουργία ενός εργαλείου, του ΕΚΑΤΑΙΟΥ ΙΙ, το οποίο θα έχει τη δυνατότητα απεικόνισης του σχήματος μιας βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν, καθώς και τη δυνατότητα εύρεσης του αντίκτυπου που επιφέρει στο σύστημα μια πιθανή αλλαγή στο σχήμα της βάσης.

Η αναπαράσταση του σχήματος της βάσης δεδομένων και των ερωτημάτων/όψεων που την προσπελαύνουν έγινε με ενιαίο τρόπο ως κατευθυνόμενος γράφος. Οι κόμβοι του γράφου αναπαριστούν τα αντικείμενα του σχήματος της βάσης δεδομένων (σχέσεις, γνωρίσματα, περιορισμοί, όψεις) και τα στοιχεία των ερωτημάτων/όψεων που προσπελαύνουν τα αντικείμενα αυτά (ερωτήματα, όψεις, γνωρίσματα ερωτημάτων, γνωρίσματα όψεων, τελεστές, σταθερές), ενώ οι ακμές αναπαριστούν τις μεταξύ τους σχέσεις και αλληλεξαρτήσεις.

Για τη μελέτη της εξέλιξης του σχήματος της βάσης προστέθηκε σημασιολογία πάνω στους κόμβους του γράφου. Συγκεκριμένα, μπορούν πάνω στους κόμβους του γράφου να οριστούν γεγονότα που αλλάζουν το σχήμα της βάσης (π.χ. «προσθήκη ενός γνωρίσματος σε μια σχέση» ή «μετονομασία μιας σχέσης») και πολιτικές που χειρίζονται συγκεκριμένα γεγονότα που αφορούν συγκεκριμένους κόμβους (π.χ. «Στην περίπτωση προσθήκης ενός γνωρίσματος στη σχέση Χ εμπόδισε την αλλαγή» ή «Στην περίπτωση μετονομασίας της σχέσης Χ προώθησε την αλλαγή»).

Βάσει αυτής της σημασιολογίας όσον αφορά στην εξέλιξη και μιας μεθοδολογίας που βασίζεται στην κατεύθυνση των ακμών που ενώνει τους κόμβους του γράφου, είναι δυνατό να βρεθεί η νέα κατάσταση των κόμβων, που δηλώνει πως αυτοί επηρεάζονται από ένα γεγονός.

Το εργαλείο προσφέρει επιπλέον την οπτικοποίηση του γράφου και το γραφικό χειρισμό του, γεγονός που το κάνει πιο φιλικό προς το χρήστη.

7.2 Μελλοντικές επεκτάσεις

Στην ενότητα αυτή δίνουμε ιδέες για επέκταση της διπλωματικής.

7.2.1 Αυτοματοποιημένη τροποποίηση του γράφου.

Μια πολύ χρήσιμη επέκταση της διπλωματικής, είναι η αυτοματοποιημένη τροποποίηση του γράφου, σε περίπτωση που ο διαχειριστής της βάσης δεδομένων αποδεχθεί ένα πιθανό σενάριο. Δηλαδή, αντί να αλλάζουν απλώς οι καταστάσεις των κόμβων δείχνοντας στο διαχειριστή της βάσης τις ενέργειες που πρέπει να γίνουν για να ενσωματωθεί η προτεινόμενη αλλαγή, να υπήρχε η δυνατότητα να αφαιρούνται και να προσθέτονται κατάλληλα κόμβοι και ακμές.

7.2.2 Εξαγωγή του γράφου σε αρχεία sql.

Σε συνδυασμό με την παραπάνω επέκταση, θα ήταν περισσότερο χρήσιμο ο τροποποιημένος γράφος ενός σεναρίου να μπορεί να αποθηκευτεί σε αρχεία τύπου sql, δηλαδή στο αρχείο ορισμού των δεδομένων της βάσης, καθώς και στο αρχείο που περιλαμβάνει τον ορισμό των όψεων και των ερωτημάτων που την προσπελαύνουν. Με αυτόν τον τρόπο θα γίνεται αυτόματα η αλλαγή στο σχήμα της βάσης δεδομένων, τις όψεις και τα ερωτήματα, ως απόκριση σε ένα γεγονός που αλλάζει το σχήμα της βάσης, αντί να αλλάζει απλώς ο γράφος που αναπαριστά το σύστημα.

8 Παράρτημα

8.1 DDL και SQL αρχεία για την ενότητα 6.

Οι πίνακες TEST και TESTB που θα εισαχθούν για τον έλεγχο του συστήματος περιγράφονται στο DDL αρχείο:

```
-- Create table
create table TEST
(
  TEST_RES_YEAR      NUMERIC(4) not null,
  TEST_RSC_CODE      VARCHAR(4) not null,
  TEST_CODE          VARCHAR(8) not null,
  TEST_AA_QUE        NUMERIC(7),
  TEST_LAST_NAME     VARCHAR(50),
  TEST_FIRST_NAME    VARCHAR(30),
  TEST_FATHERNAME    VARCHAR(25),
  TEST_MOTHERNAME    VARCHAR(25),
  TEST_BIRTH_YR      NUMERIC(4),
  TEST_IDCARD        VARCHAR(8),
  TEST_AFM           VARCHAR(10),
  TEST_ADDRESS       VARCHAR(50),
  TEST_ZIPCODE       VARCHAR(5),
  TEST_TEL           VARCHAR(15),
  TEST_FAX           VARCHAR(15),
  TEST_LEGALPERS     VARCHAR(80),
```

```
  TEST_NMAX          NUMERIC(7),
  TEST_NMIN          NUMERIC(7),
  TEST_LAND          NUMERIC(9,1),
  TEST_UTILLAND      NUMERIC(9,1),
  TEST_SM_CODE       VARCHAR(10),
  TEST_LEV           VARCHAR(2),
  TEST_CLA_CODE      VARCHAR(2),
  TEST_CREATED_BY    VARCHAR(30),
  TEST_CHANGED_BY    VARCHAR(30),
  TEST_DATE_CREATED  DATE,
  TEST_DATE_CHANGED  DATE,
  TEST_GEOCODE       VARCHAR(8) not null,
  TEST_LAY_CODE      VARCHAR(4),
  SM_CHA_CODE       VARCHAR(4),
  TEST_TOTSUM        NUMERIC(10,2),
  TEST_OTHNAME       VARCHAR(80),
  TEST_MEMBER        NUMERIC(1),
  TEST_PRACTICE      NUMERIC(1),
  TEST_REMARKS       VARCHAR(80),
  TEST_QUED          NUMERIC(7),
  TEST_AA_QUED       NUMERIC(7),
  TEST_BASTRAINAGR   NUMERIC(1),
  TEST_TRAINAGR      NUMERIC(1),
  TEST_OPERATION     VARCHAR(2),
  TEST_SECTION       VARCHAR(5),
  TEST_FINTYPE       NUMERIC(6),
  TEST_PROPERTY      VARCHAR(2),
  TEST_LEGAL         VARCHAR(2),
  TEST_COMMERCIAL    NUMERIC(1),
  TEST_ACCBOOK       NUMERIC(1),
  TEST_PERSONAL      NUMERIC(1),
  TEST_RELATION      VARCHAR(2),
  TEST_SALE          NUMERIC(1),
  TEST_ASSOCIATION   NUMERIC(1),
  TEST_OGA           NUMERIC(1),
```

```
  TEST_DEPT          VARCHAR(4),
  TEST_FLG           VARCHAR(1),
  TEST_PERC2         NUMERIC(4,1),
  TEST_PERC3         NUMERIC(4,1),
  TEST_YPA           VARCHAR(4),
  TEST_YPACODE       VARCHAR(10),
  TEST_AA_TMPQUE     NUMERIC(4),
  TEST_BLOCK         NUMERIC(5),
  TEST_NOMOS         VARCHAR(4),
  TEST_DIMOS         VARCHAR(8),
  TEST_TSGM          NUMERIC(10,2),
  TEST_FEME          NUMERIC(3),
  TEST_SURFCODE      NUMERIC(9),
  TEST_LAND_OTH      NUMERIC(9,1),
  TEST_UTILLAND_OTH  NUMERIC(9,1),
  TEST_TOTAL         NUMERIC(9,2),
  TEST_LAYEURO_CODE  VARCHAR(4),
  TEST_GEO_THESH     VARCHAR(8),
  TEST_MET           VARCHAR(1),
  TEST_GEO_MET       VARCHAR(8),
  TEST_THESH_ADRESS  VARCHAR(40),
  TEST_SYDEL         NUMERIC(10,5),
  TEST_STREET_NUMBER VARCHAR(6),
  constraint  TEST_PK  primary  key  (TEST_RES_YEAR,  TEST_RSC_CODE,
TEST_CODE)
);



-- Create table
create table TESTB
(
  TEST_RES_YEAR          NUMERIC(4) not null,
  TEST_RSC_CODE          VARCHAR(4) not null,
  TEST_CODE              VARCHAR(8) not null,
```

```
  TEST_CREATED_BY        VARCHAR(30),
  TEST_CHANGED_BY        VARCHAR(30),
  TEST_DATE_CREATED      DATE,
  TEST_DATE_CHANGED      DATE,
  TEST_NUMBER_NEW        NUMERIC(2),
  TEST_CURRENT_STATE     VARCHAR(1),
  TEST_AGR_OCCUPATION    VARCHAR(1),
  TEST_NEW_GEOCODE       VARCHAR(8),
  TEST_TRAINING          VARCHAR(1),
  TEST_TRAINING_LEVEL    VARCHAR(1),
  TEST_CONSUMPTION       VARCHAR(1),
  TEST_SALE              VARCHAR(1),
  TEST_OTHER_POSITION    VARCHAR(1),
  TEST_OTHER_GEOCODE     VARCHAR(8),
  TEST_OTHER_AREA        NUMERIC(9,2),
  TEST_OTHER_STAVLOI     NUMERIC(9,2),
  TEST_CURRENT_AREA      NUMERIC(9,2),
  TEST_CURRENT_STAVLOI   NUMERIC(9,2),
  TEST_FACE              VARCHAR(1),
  TEST_DURATION          VARCHAR(2),
  TEST_DATE              DATE,
  TEST_COOPERATION       VARCHAR(1),
  TEST_NOTES             VARCHAR(4000),
  TEST_GEOCODE           VARCHAR(8),
  TEST_NEW_LAST_NAME     VARCHAR(50),
  TEST_NEW_FIRST_NAME    VARCHAR(30),
  TEST_NEW_FATHERNAME    VARCHAR(25),
  TEST_NEW_BIRTH_YR      NUMERIC(4),
  TEST_NEW_IDCARD        VARCHAR(8),
  TEST_NEW_LEGALPERS     VARCHAR(80),
  TEST_NEW_AFM           VARCHAR(10),
  TEST_NEW_STREET        VARCHAR(50),
  TEST_NEW_STREET_NUMBER VARCHAR(4),
  TEST_NEW_ZIPCODE       VARCHAR(5),
  TEST_NEW_TEL           VARCHAR(15),
```

```
  TEST_BIO_ZWA_FLG       NUMERIC(1),
  constraint  TESTB_PK  primary  key  (TEST_RES_YEAR,  TEST_RSC_CODE,
TEST_CODE),
  constraint  TESTB_FK  foreign  key  (TEST_RES_YEAR,  TEST_RSC_CODE,
TEST_CODE)
  references TEST (TEST_RES_YEAR, TEST_RSC_CODE, TEST_CODE)
);
```

Η όψη TEST_V και τα ερωτήματα που προσπελαύνουν τους παραπάνω πίνακες περιγράφονται στο SQL αρχείο:

```
CREATE VIEW TEST_V AS
SELECT
        S.Test_res_year,
        S.Test_rsc_code,
        S.Test_code,
        S.Test_last_name,
        S.Test_first_name,
        S.Test_fathername,
        S.Test_mothername,
        S.Test_birth_yr,
        S.Test_idcard,
        S.Test_afm,
        S.Test_address,
        S.Test_zipcode,
        S.Test_tel,
        S.Test_legalpers,
        S.Test_nmax,
        S.Test_nmin,
        S.Test_sm_code,
        S.Test_created_by,
        S.Test_changed_by,
        S.Test_date_created,
        S.Test_date_changed,
        S.Test_geocode,
        S.Test_lay_code,
        S.sm_cha_code,
```

```
        S.Test_member,
        S.Test_qued,
        S.Test_aa_qued,
        S.Test_section,
        S.Test_legal,
        S.Test_personal,
        S.Test_relation,
        S.Test_dept,
        S.Test_flg,
        S.Test_ypa,
        S.Test_ypacode,
        S.Test_block,
        S.Test_nomos,
        S.Test_dimos,
        S.Test_tsgm,
        s.Test_fintype,
        S.Test_feme,
        S.Test_surfcode,
        S.Test_sydel,
        S.Test_street_number,
        s.Test_met,
        s.Test_totsum,
        D.TEST_NUMBER_NEW,
        D.TEST_CURRENT_STATE,
        D.TEST_AGR_OCCUPATION,
        D.TEST_NEW_GEOCODE,
        D.TEST_TRAINING,
        D.TEST_TRAINING_LEVEL,
        D.TEST_CONSUMPTION,
        D.TEST_SALE,
        D.TEST_OTHER_POSITION,
        D.TEST_OTHER_GEOCODE,
        D.TEST_OTHER_AREA,
        D.TEST_OTHER_STAVLOI,
        D.TEST_CURRENT_AREA,
```

```
        D.TEST_CURRENT_STAVLOI,
        D.TEST_FACE,
        D.TEST_DURATION,
        D.TEST_DATE,
        D.TEST_COOPERATION,
        D.TEST_NOTES,
        D.TEST_NEW_LAST_NAME,
        D.TEST_NEW_FIRST_NAME,
        D.TEST_NEW_FATHERNAME,
        D.TEST_NEW_BIRTH_YR,
        D.TEST_NEW_IDCARD,
        D.TEST_NEW_LEGALPERS,
        D.TEST_NEW_AFM,
        D.TEST_NEW_STREET,
        D.TEST_NEW_STREET_NUMBER,
        D.TEST_NEW_ZIPCODE,
        D.TEST_NEW_TEL,
        D.TEST_BIO_ZWA_FLG
  From TEST S, TESTB D
  Where S.Test_rsc_code = D.Test_rsc_code
    and S.Test_res_year = D.Test_res_year
    and S.Test_code     = D.Test_CODE;


 ---------------QUERY1----------------------------;
 Select  Test_geocode, count(Test_code)
  From spr_samples s
  Where Test_rsc_code = '01'
    and Test_res_year = 2005
    and s.Test_nomos  ='24'
  Group By Test_geocode;


--------------------QUERY 2----------------------;
select s.Test_res_year,
```

```
        s.Test_rsc_code,
        s.Test_code,
        s.Test_legal
        Test_legal,
      s.Test_personal Test_personal,
      s.Test_member Test_member,
      s.Test_relation Test_relation,
      s.Test_section kod_str,
        s.Test_sydel,
      s.Test_geocode,
        s.Test_training Test_training,
      s.Test_training_level Test_training_level,
      s.Test_consumption Test_consumption,
      s.Test_sale     Test_sale,
        Test_BIO_ZWA_FLG BIO_ZWA
      from TEST_v s
      where s.Test_rsc_code     = '01'
      and s.Test_res_year     = 2005
        and sm_cha_code between 1 and 8     ;


----------------------------QUERY 3--------------------;
SELECT                   TEST_RES_YEAR,
                         TEST_RSC_CODE,
                  TEST_CODE,
                         TEST_LAST_NAME,
                         TEST_FIRST_NAME,
                         TEST_FATHERNAME,
                         TEST_BIRTH_YR,
                         TEST_IDCARD,
                         TEST_LEGALPERS,
                         TEST_AFM,
                         TEST_ADDRESS,
                         TEST_STREET_NUMBER,
                         TEST_ZIPCODE,
```

```
                        TEST_TEL,
                        TEST_NMAX,
                        TEST_NMIN,
                        TEST_SM_CODE,
                        TEST_GEOCODE,
                        TEST_NOMOS,
                  TEST_LAY_CODE,
                  SM_CHA_CODE,
                        TEST_MEMBER ,
                        TEST_QUED ,
                        TEST_AA_QUED ,
                        TEST_SECTION ,
                        TEST_LEGAL ,
                        TEST_PERSONAL ,
                        TEST_RELATION ,
                        TEST_DEPT,
                        TEST_FLG,
                        TEST_CREATED_BY,
                        TEST_DATE_CREATED,
                        TEST_SYDEL
FROM TEST
where Test_rsc_code     = '01'
and Test_res_year     = 2005
and Test_code = '01010101';


------------------------QUERY 4----------------------------;
Select s.Test_geocode ,
        s.Test_code,
        S.TEST_LAST_NAME Lastname


From Test_v s
where s.Test_res_year=2005
and   s.Test_rsc_code='01'
AND   S.Test_nomos  = '01'
```

9 Βιβλιογραφία

[Bane87] J. Banerjee et al. Semantics and implementation of schema evolution in object-oriented databases. In Proceedings of ACM SIGMOD. San Francisco, California, May 1987.

[Bell02] Z. Bellahsene. Schema evolution in data warehouses. In Knowledge and Information Systems, 4 (2002): p.283-304.

[Bern03] P. Bernstein. Applying Model Management to Classical Meta Data Problems. In Proc. of the CIDR Conference, Asilomar, California. January 2003.

[CadL03] Rogers Cadenhead, Laura Lemay, Πλήρες Εγχειρίδιο της Java 2, Εκδότης Μ.Γκιούρδας, 2003.

[Gall02] David Gallardo, Getting started with the Eclipse Platform, 1 Νοεμβρίου 2002, http://www.ibm.com/developerworks/library/os-ecov/.

[GMRR01] A. Gupta, I. S. Mumick, J. Rao, K. A. Ross. Adapting materialized views after redefinitions: Techniques and a performance study. In Information Systems, 26 (2001), p.323-362.

[LiCC94] C.T. Liu, P.K. Chrysanthis, S.K. Chang. Database schema evolution through the specification and maintenance of changes on entities and relationships. In Proceedings of the International Conference on Entity-Relationship Approach (ER '94), p.132-151. Manchester, U.K., Dec 1994.

[Lourid] Περίπτωση Χρήσης Ολοκληρωμένου Περιβάλλοντος Ανάπτυξης Eclipse http://www.dmst.aueb.gr/louridas/notes/dais/tools/ar01s14.html.

[Meln04] S. Melnik. Generic Model Management - Concepts and Algorithms. LNCS 2967 - Springer-Verlag Berlin 2004.

[MFWB] Joshua O'Madadhain, Danyel Fisher, Scott White, Yan-Biao Boey. The JUNG (Java Universal Network/Graph) Framework, http://www.datalab.uci.edu/papers/JUNG_tech_report.html.

[MoDo96] M. Mohania, D. Dong. Algorithms for adapting materialized views in data warehouses. In Proceedings of 8th international symposium on cooperative database systems for advanced applications (CODAS '96), p.309-316. Kyoto, Japan, Dec 1996.

[NiLR98] A. Nica, A. J. Lee, E. A. Rundensteiner. The CSV algorithm for view synchronization in evolvable large-scale information systems. In Proceedings of International Conference on Extending Database Technology (EDBT '98). Lectures notes in computer science, Springer, p.359-373. Valencia, Spain, Mar 1998.

[PKVV05] G. Papastefanatos, K. Kyzirakos, P. Vassiliadis, Y. Vassiliou. Hecataeus: A Framework for Representing SQL Constructs as Graphs. In 10th International Workshop on Exploring Modeling Methods for Systems Analysis and Design – EMMSAD '05 (in conjunction with CAISE' 05)

[PVSV07] G. Papastefanatos, P. Vassiliadis, A. Simizis, Y. Vassiliou. What-if analysis for Data Warehouse Evolution in 9th International Conference on Data Warehousing and Knowledge Discovery (Dawak 07), Regensburg, Germany, 3-7 September, 2007.

[RaRu95] Y. G. Ra, E. A. Rundensteiner. A transparent object-oriented schema change approach using view evolution. In Proceedings of the Eleventh International Conference on Data Engineering (ICDE '95), p.165-172, Taipei, Taiwan, Mar 1995.

[Rodd00] J.F. Roddick, L. Al-Jadir, L.E. Bertossi, M. Dumas, F. Estrella, H. Gregersen, K. Hornsby, J. Lufter, F. Mandreoli, T. Männistö, E. Mayol, L. Wedemeijer. Evolution and Change in Data Management - Issues and Directions. SIGMOD Record 29(1): p.21-25 (2000).

[Rodd95] J.F. Roddick. A survey of schema versioning Issues for database systems. In Information Software Technology, 37(7): p.383-393.

[RuLN97] E. A. Rundensteiner, A. J. Lee, A. Nica. On preserving views in evolving environments. In Proceedings of the 4th KRDB Workshop, p.13.1–13.11. Athens, Greece, Aug 1997.

[VeMP04] Y. Velegrakis, R. Miller, L. Popa. Preserving Mapping consistency under schema changes. VLDB Journal (2004) 13:274-293

[Wiki07] Wikipedia, 25 Νοεμβρίου 2007, http://en.wikipedia.org/wiki/Java_programming_language.

[Zica91] R. Zicari. A framework for schema update in an object-oriented database system. In Proceedings of Conference on Data Engineering (ICDE '91). Kobe, Japan, Apr 1991.