ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΣΤΗ ΔΙΟΙΚΗΣΗ ΚΑΙ ΟΙΚΟΝΟΜΙΚΗ ΤΩΝ ΤΗΛΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΔΙΚΤΥΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη Συστήματος Οπτικοποίησης Διανυσματικών Χαρτών

Γεώργιος Κ. Ρεπάσος

Γεώργιος Δ. Σταματούκος

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Ευστάθιος Χατζηευθυμιάδης,** Καθηγητής ΕΚΠΑ
Βασίλειος Τσέτσος, Υποψήφιος Διδάκτωρ ΕΚΠΑ
Κωνσταντίνος Κολομβάτσος, Υποψήφιος Διδάκτωρ ΕΚΠΑ

ΑΘΗΝΑ

ΑΠΡΙΛΙΟΣ 2008

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη Συστήματος Οπτικοποίησης Διανυσματικών Χαρτών

Γεώργιος Κ. Ρεπάσος
Α.Μ.: ΜΟΠ70

Γεώργιος Δ. Σταματούκος
Α.Μ.: ΜΟΠ64

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Ευστάθιος Χατζηευθυμιάδης,** Καθηγητής ΕΚΠΑ
Βασίλειος Τσέτσος, Υποψήφιος Διδάκτωρ ΕΚΠΑ
Κωνσταντίνος Κολομβάτσος, Υποψήφιος Διδάκτωρ ΕΚΠΑ

Απρίλιος 2008

ΠΕΡΙΛΗΨΗ

Σε αυτή τη διπλωματική εργασία δημιουργήθηκε μια εφαρμογή πλοήγησης σε εσωτερικούς χώρους με χρήση open source λογισμικού. Η απεικόνιση των εσωτερικών χώρων έγινε με χρήση χαρτών σε ανυσματική (vector) μορφή. Για την απεικόνιση των χαρτών του κτιρίου χρησιμοποιήθηκε το πρότυπο svg. Η εφαρμογή ακολουθεί την αρχιτεκτονική client – server. Ο τρόπος υλοποίησής της είναι platform independent καθώς ως client μπορεί να είναι οποιαδήποτε συσκευή που υποστηρίζει javascript και svg (π.χ.pc, φορητός υπολογιστής, pda, κινητό τηλέφωνο).

Στόχος της εφαρμογής είναι η πλοήγηση σε εσωτερικούς χώρους με χρήση κινητών συσκευών χωρίς την απαίτηση εγκατάστασης εξειδικευμένων προγραμμάτων που υπόκεινται στο καθεστώς των πνευματικών δικαιωμάτων.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Πλοήγηση Εσωτερικών Χώρων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: Γεωγραφικά Συστήματα Υπηρεσιών, Υπηρεσίες Βασισμένες στη Θέση, SVG, Mapserver, ανυσματικά δεδομένα

ABSTRACT

In this Master Thesis an indoor navigation application, using open source software, was created. The visualization of indoor spaces was implemented by maps in vector format. In particular the maps are viewed in svg format. The application uses a client – server architecture. The application is platform independent as every device (pc, notebook, pda, mobile phone, etc) that supports javascript and svg can be used as a client.

The aim of the application is indoor navigation using mobile devices without the need of specific proprietary software.

SUBJECT AREA: Indoor Navigation

KEYWORDS: GIS, LBS, SVG, Mapserver, Vector Data

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ

Η εργασία αυτή διενεργήθηκε στα πλαίσια του μεταπτυχιακού προγράμματος σπουδών «Οικονομική και Διοίκηση Τηλεπικοινωνιακών Δικτύων» του τμήματος Πληροφορικής και Τηλεπικοινωνιών υπό την επίβλεψη του επίκουρου καθηγητή Ευστάθιου Χατζηευθυμιάδη και των υποψήφιων διδακτόρων Βασιλείου Τσέτσου και Κωνσταντίνου Κολομβάτσου. Η εργασία πραγματοποιήθηκε με εξίσου ενασχόληση των δύο μεταπτυχιακών φοιτητών του προγράμματος και μπορεί να αποτελέσει τμήμα μιας ευρύτερης εφαρμογής πλοήγησης σε εσωτερικούς χώρους. Η καινοτομία της αφορά στη χρήση ανυσματικών εικόνων για την απεικόνιση του χώρου σε web εφαρμογές με χρήση open source λογισμικού.

Ευχαριστούμε τον επίκουρο καθηγητή κ. Ευστάθιο Χατζηευθυμιάδη και τους υποψήφιους διδάκτορες Βασίλειο Τσέτσο και Κωνσταντίνο Κολομβάτσο για την πολύτιμη βοήθεια και το ενδιαφέρον τους.

ΚΕΦΑΛΑΙΟ 1

ΥΠΗΡΕΣΙΕΣ ΒΑΣΙΣΜΕΝΕΣ ΣΤΗ ΘΕΣΗ

1.1. LBS – Ορισμός

Τα κινητά τηλέφωνα και το Διαδίκτυο έχουν φέρει την επανάσταση στην επικοινωνία και στον τρόπο ζωή των ανθρώπων. Ένας αυξανόμενος αριθμός κινητών τηλεφώνων και PDAs επιτρέπουν στους ανθρώπους να έχουν πρόσβαση στο Διαδίκτυο όποτε θέλουν και από οποιοδήποτε μέρος. Από το Διαδίκτυο μπορούν αφενός να λάβουν άμεσα πληροφορίες για γεγονότα που τους ενδιαφέρουν (κινηματογράφος, συναυλίες, θέατρα) και αφ' ετέρου πληροφορίες για τους χώρους, όπου αυτά τα γεγονότα θα πραγματοποιηθούν (χάρτες πόλεων, εστιατόρια, μουσεία, νοσοκομεία).

Έστω ότι κάποιος θέλει να πάει για φαγητό σε ένα εστιατόριο και επομένως ψάχνει ένα εστιατόριο στο Διαδίκτυο. Για να αποφύγει την περίπτωση να πάρει ως αποτέλεσμα αναζήτησης όλες τις ιστοσελίδες εστιατορίων στον κόσμο πρέπει να περιορίσει την αναζήτηση του προσθέτοντας παραπάνω κριτήρια αναζήτησης. Μια καλή επιλογή είναι η προσθήκη της πόλης όπου ο κινητός χρήστης βρίσκεται (θέση), ο πραγματικός χρόνος (βράδυ) ή ένας ειδικός τύπος εστιατορίου (κινέζικο ή ελληνικό).

Μια τέτοια αναζήτηση εστιατορίων όσον αφορά τη θέση και το χρόνο μπορεί να γίνει με τη χρήση υπηρεσιών βασισμένων στη θέση (Location Based Services - LBS). Κατά συνέπεια, ένας ορισμός των LBS είναι:

LBS Ορισμός 1:

Οι LBS είναι υπηρεσίες πληροφοριών προσβάσιμες από κινητές συσκευές μέσω ασύρματου δικτύου, οι οποίες αξιοποιούν τη δυνατότητα χρησιμοποίησης της θέση τους.

Ένας παρόμοιος ορισμός των LBS δίνεται από το διεθνές OpenGeospatial Consortium[1]:

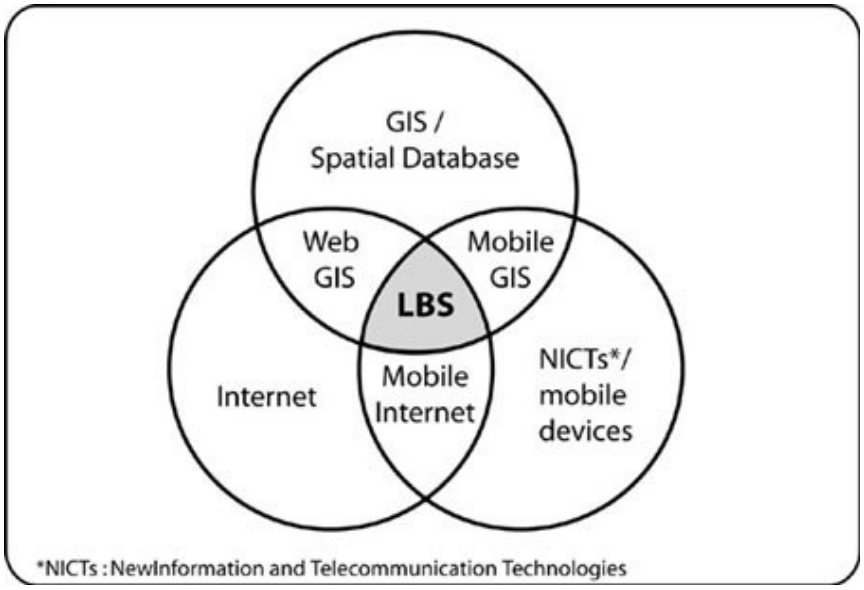
LBS Ορισμός 2:

Οι LBS είναι μία wireless-IP υπηρεσία που χρησιμοποιεί γεωγραφικές πληροφορίες για να εξυπηρετήσει έναν κινητό χρήστη. Συνεπώς ως LBS θεωρείται οποιαδήποτε υπηρεσία εφαρμογής που εκμεταλλεύεται τη θέση ενός κινητού τερματικού.

Αυτοί οι ορισμοί περιγράφουν τις LBS ως την τομή τριών τεχνολογιών (Εικόνα 1). Δημιουργείται από τις Καινούριες Τεχνολογίες Επικοινωνιών και Πληροφοριών (New

Information and Communication Technologies - NICTS), όπως για παράδειγμα το κινητό σύστημα τηλεπικοινωνιών και οι hand held συσκευές, από το Διαδίκτυο και από τα Γεωγραφικά Συστήματα Πληροφοριών (Geographic Information Systems) με τις χωρικές βάσεις δεδομένων [2].

Από ιστορική άποψη οι πληροφορίες που βασίζονται στη θέση δεν είναι κάτι καινούριο που εμφανίστηκε με την εφεύρεση των κινητών τηλεφώνων. Ο Espinoza [3] τόνισε ότι οι συγκεκριμένες πληροφορίες θέσης μεταφέρονται επίσης σε μια επικοινωνία πρόσωπο με πρόσωπο με σημειώσεις post-it ή με graffiti. Επίσης οι μέθοδοι για την ενημέρωση μιας τοπικής κοινωνίας μπορούν να είναι αφίσες (π.χ. των συναυλιών στην πόλη) ή απλά σημάδια κυκλοφορίας, οι οποίες υποβάλλουν τις πληροφορίες πλοήγησης. Σ' αυτές τις περιπτώσεις η επικοινωνία είναι μονόδρομη. Οι LBS δίνουν τη δυνατότητα αμφίδρομης επικοινωνίας και αλληλεπίδρασης. Επομένως ο χρήστης αναφέρει στον πάροχο των υπηρεσιών δεδομένα που σχετίζονται με αυτόν, όπως το είδος των πληροφοριών που χρειάζεται, τις προτιμήσεις του και τη θέση του. Αυτό βοηθά τον πάροχο τέτοιων υπηρεσιών θέσης να παραδώσει πληροφορίες προσαρμοσμένες στις ανάγκες των χρηστών.



Εικόνα 1: Τα LBS σαν τομή τεχνολογιών

1.2. Τμήματα των LBS

Τα βασικά στοιχεία της τηλεπικοινωνιακής υποδομής ενός LBS συστήματος φαίνονται στην Εικόνα 2 και είναι τα εξής:

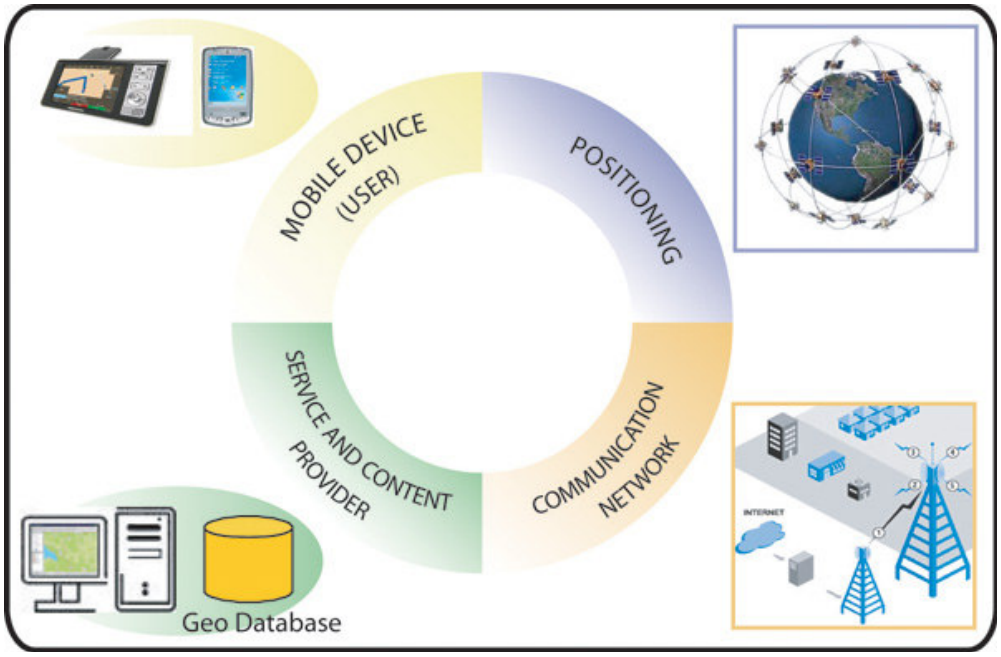
Κινητές Συσκευές: Ένα εργαλείο για το χρήστη για να ζητήσει τις αναγκαίες πληροφορίες. Τα αποτελέσματα μπορούν να δοθούν με την ομιλία, χρησιμοποιώντας εικόνες, κείμενο κτλ. Τέτοιες συσκευές συνήθως είναι τα PDAs, τα κινητά τηλέφωνα και τα laptops. Επίσης μια τέτοια συσκευή θα μπορούσε επίσης να είναι μια μονάδα πλοήγησης του αυτοκινήτου.

Δίκτυο Επικοινωνίας: Το δεύτερο τμήμα είναι το ασύρματο δίκτυο που μεταφέρει τα στοιχεία και τα αιτήματα των χρηστών από το κινητό τερματικό στον πάροχο της υπηρεσίας και έπειτα τη ζητούμενη πληροφορία πίσω στο χρήστη.

Τμήμα Εντοπισμού Θέσης: Για την επεξεργασία ενός αιτήματος συνήθως πρέπει να καθοριστεί η θέση των χρηστών. Η θέση των χρηστών μπορεί να ληφθεί είτε με τη χρησιμοποίηση του δικτύου κινητής επικοινωνίας ή με τη χρησιμοποίηση του Παγκόσμιου Συστήματος Εντοπισμού (Global Positioning System). Ο καθορισμός της θέσης μπορεί επιπλέον να πραγματοποιηθεί με χρήση σταθμών WLAN, active badges ή radio beacons. Οι τελευταίες μέθοδοι προσδιορισμού θέσης μπορούν να χρησιμοποιηθούν για την πλοήγηση σε εσωτερικούς χώρους (indoor navigation), όπως για παράδειγμα σε ένα μουσείο.

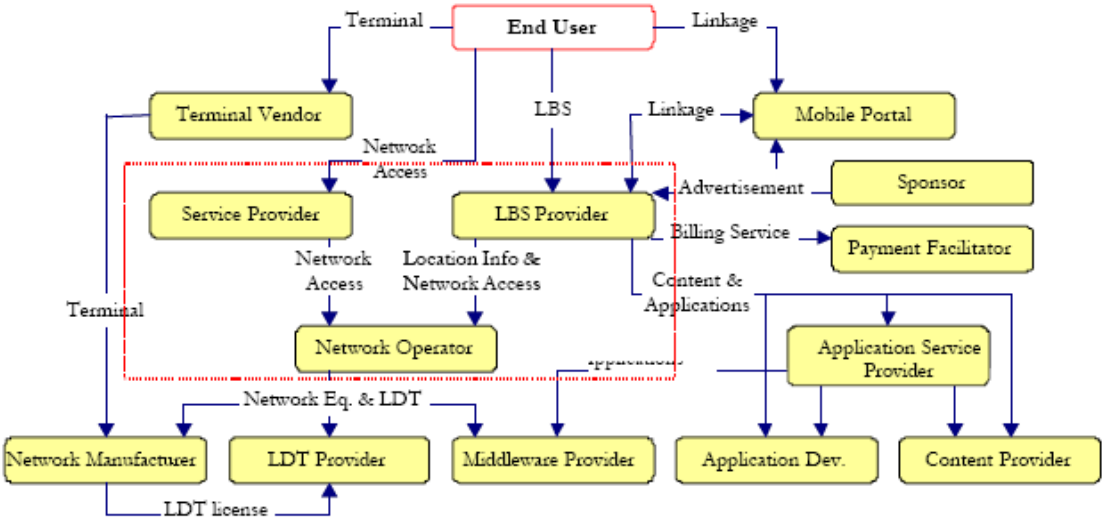
Πάροχος υπηρεσίας και εφαρμογής: Ο πάροχος των υπηρεσιών αυτών LBS προσφέρει έναν αριθμό διαφορετικών υπηρεσιών στο χρήστη και είναι αρμόδιος για την επεξεργασία των αιτημάτων. Τέτοιες υπηρεσίες προσφέρουν τον υπολογισμό της θέσης, την εύρεση μιας διαδρομής, την έρευνα των πληροφοριών του χρυσού οδηγού όσον αφορά τη θέση ή την έρευνα συγκεκριμένων πληροφοριών για αντικείμενα ενδιαφέροντος των χρηστών (π.χ. ένα πουλί σε ένα ζωολογικό πάρκο) και ούτω καθ' εξής.

Πάροχος περιεχομένου και δεδομένων: Οι πάροχοι περιεχομένου και δεδομένων συνήθως δεν αποθηκεύουν και διατηρούν όλες τις πληροφορίες που μπορεί να ζητηθούν από τους χρήστες. Επομένως τα βασικά γεωγραφικά δεδομένα και τα δεδομένα πληροφοριών θέσης ζητούνται συνήθως από την αρχή διατήρησης των δεδομένων αυτών (π.χ. υπηρεσίες χαρτογράφησης) ή επιχειρήσεις (π.χ. χρυσός οδηγός, επιχειρήσεις κυκλοφορίας)



Εικόνα 2: Τα βασικά συστήματα ενός LBS συστήματος

Στην Εικόνα 3 φαίνεται πιο αναλυτικά η αλληλεπίδραση όλων των τμημάτων ενός LBS συστήματος και η ιεράρχησή τους.



Εικόνα 3: Αλληλεπίδραση τμημάτων LBS

1.3. Χρησιμότητα των LBS συστημάτων

Η ιδέα πίσω από τα LBS συστήματα και η χρησιμότητα τους είναι να απαντούν ερωτήσεις του τύπου *Πού είμαι; Πού είναι οι φίλοι μου; Τι υπάρχει εδώ γύρω μου;*. Όταν σχεδιάζεται ένα LBS σύστημα οι ανάγκες του χρήστη πρέπει να ικανοποιηθούν προκειμένου να καταστήσουν τις υπηρεσίες, που παρέχονται, χρήσιμες.

Όταν κάποιος βρίσκεται σε ένα περιβάλλον, το οποίο του είναι άγνωστο, η συμπεριφορά του και οι ανάγκες του είναι κατά ένα μεγάλο μέρος προβλέψιμες, είτε

βρίσκεται στη χώρα του ή στο εξωτερικό, είτε βρίσκεται σ' ένα όχημα ή περπατάει. Αυτά που συνήθως χρειάζεται είναι ένα εστιατόριο, ένα φαρμακείο, μια τράπεζα, μια στάση μετρό κτλ. Όταν κάποιος βρίσκεται στο εξωτερικό, οι ανάγκες του είναι μεγαλύτερες: να βρει τα τοπικά τουριστικά αξιοθέατα, τον τρόπο με τον οποίο θα τα προσεγγίσει, να βρει ένα ξενοδοχείο που θα καλύπτει τις απαιτήσεις του και ένα ανταλλακτήριο συναλλάγματος. Κατά την οδήγηση, ενδεχομένως να υπάρξουν κι άλλες ανάγκες, όπως η βοήθεια για την εύρεση μιας διαδρομής σε μια άγνωστη πόλη. Σήμερα, ένας ανεπαρκώς προετοιμασμένος ταξιδιώτης (που δεν συμβουλεύεται το Διαδίκτυο, δεν αγοράζει ένα τουριστικό οδηγό, δεν παίρνει τις απαραίτητες πληροφορίες από τα ξενοδοχεία, κτλ) ξοδεύει πολύ χρόνο, και δεν θα βοηθηθεί αρκετά από το κινητό τηλέφωνό του[4].

1.4. Ταξινόμηση LBS

Οι LBS μπορούν να ταξινομηθούν *σε reactive* και *proactive LBS*. Οι *reactive* LBS ενεργοποιούνται πάντα από το χρήστη. Η αλληλεπίδραση μεταξύ LBS και χρήστη είναι κατά προσέγγιση η ακόλουθη: ο χρήστης καλεί αρχικά την υπηρεσία και ξεκινάει ένα session, είτε μέσω μιας κινητής συσκευής ή με ένα desktop PC. Στη συνέχεια ζητά ορισμένες εφαρμογές ή πληροφορίες, και η υπηρεσία συγκεντρώνει τα στοιχεία της θέσης (είτε της δικής του, είτε κάποιου άλλου στόχου), τα επεξεργάζεται, και επιστρέφει το βασισμένο στη θέση αποτέλεσμα στο χρήστη, παραδείγματος χάριν, ένα κατάλογο κοντινών εστιατορίων. Αυτός ο κύκλος αιτήματος/απάντησης μπορεί να επαναληφθεί αρκετές φορές πριν την ολοκλήρωση του session. Κατά συνέπεια, τα reactive LBS χαρακτηρίζονται από μια σύγχρονη αλληλεπίδραση μεταξύ του χρήστη και της υπηρεσίας. Οι *proactive* LBS ενεργοποιούνται αυτόματα μόλις ένα προκαθορισμένο γεγονός θέσης εμφανίζεται, παραδείγματος χάριν, εάν ο χρήστης εισέλθει, προσεγγίσει, ή φύγει από ένα σημείο ενδιαφέροντος. Για παράδειγμα, θεωρήστε έναν ηλεκτρονικό τουριστικό οδηγό που ειδοποιεί τους τουρίστες μέσω SMS όταν πλησιάζουν ένα αξιοθέατο. Κατά συνέπεια, οι *proactive* υπηρεσίες δεν ενεργοποιούνται από τον χρήστη, αλλά η αλληλεπίδραση μεταξύ τους συμβαίνει ασύγχρονα. Σε αντίθεση με τις reactive LBS, που ο χρήστης εντοπίζεται μόνο μια φορά, οι proactive LBS αναζητούν το χρήστη συνέχεια προκειμένου να ανιχνεύουν τα γεγονότα που λαμβάνουν χώρα κοντά στη θέση στην οποία αυτός βρίσκεται[5].

1.4.1. Ενέργειες του χρήστη στα reactive LBS

Μια δραστηριότητα είναι μια ακολουθία ενεργειών που διευθύνονται από κάποιο πρόσωπο που στοχεύει στην επίτευξη ενός ορισμένου στόχου. Ένας τέτοιος στόχος θα

μπορούσε να είναι η λύση ενός προβλήματος ή μιας εργασίας. Όταν κάποιος βρίσκεται σε κίνηση οι στόχοι είναι, για παράδειγμα, ο προσανατολισμός, η εύρεση προσώπων ή της διαδρομής προς ένα αντικείμενο. Οι δραστηριότητες κατά τη διάρκεια της κίνησης, αυτής συχνά αφορούν σχετικές ενέργειες ενσωματωμένες στο χώρο. Αυτές οι ενέργειες οδηγούνται από ερωτήσεις ή επιθυμίες των χρηστών. Έχουν προσδιοριστεί πέντε στοιχειώδεις ενέργειες όσον αφορά τις ανάγκες των χρηστών για γεωγραφικές πληροφορίες. Η προφανέστερη ερώτηση είναι να γνωρίζει ο ίδιος ο χρήστης που είναι σε σχέση με κάτι ή κάποιον άλλο **(locating)**. Οι χρήστες μπορούν να ψάχνουν για πρόσωπα, αντικείμενα ή γεγονότα **(searching)** και να ζητήσουν τη διαδρομή προς κάποια τοποθεσία **(navigation)**. Κάτι άλλο που θα μπορούσαν να ζητήσουν είναι οι ιδιότητες και τα χαρακτηριστικά μιας τοποθεσίας **(identifying)** ή τα γεγονότα κοντά σε μια ορισμένη τοποθεσία **(checking)**. Αυτό που πρέπει να τονισθεί είναι ότι το checking χρησιμοποιεί όχι μόνο τις γεωπληροφορίες αλλά περιλαμβάνει επίσης το χρόνο, δεδομένου ότι αναφέρεται σε καταστάσεις οντοτήτων όπως επίσης και σε γεγονότα.

1.5. Εφαρμογές LBS

Τα σενάρια που παρουσιάζονται παρακάτω χωρίζονται σε επιχειρηματικές και κρατικές πρωτοβουλίες. Οι πρώτες πραγματοποιούνται από τους παρόχους υπηρεσιών για να αυξήσουν την ελκυστικότητα των δικτύων τους και των υπηρεσιών δεδομένων τους και έτσι να αυξήσουν τα μέσα έσοδα ανά χρήστη. Οι κρατικές πρωτοβουλίες από την άλλη, εισάγονται από τις κυβερνήσεις για την υποστήριξη ή την πραγματοποίηση διοικητικών στόχων.

1.5.1. Επιχειρηματικές πρωτοβουλίες

Το κύριο κίνητρο για την προσφορά LBS είναι η αύξηση του κέρδους πουλώντας πληροφορίες θέσης σε τρίτους και προσφέροντας υπηρεσίες που προσαρμόζονται στις ειδικές ανάγκες των κινητών χρηστών. Ένας πάροχος μπορεί να πραγματοποιήσει και να προσφέρει LBS με δική του πρωτοβουλία ή μπορεί να δημιουργήσει συνεργασίες με άλλες επιχειρήσεις, για παράδειγμα, από το εμπόριο ή την αυτοκινητιστική βιομηχανία, και να πραγματοποιεί και να προσφέρει τις υπηρεσίες εξ ονόματος τους.

1.5.1.1 Υπηρεσίες Πλήγησης

Οι υπηρεσίες πλοήγησης (Navigation Services) είναι βασισμένες στους κινητούς χρήστες που χρειάζονται οδηγίες δρομολόγησης στην τρέχουσα γεωγραφική θέση τους. Η δυνατότητα του κινητού δικτύου να εντοπίσει την ακριβή θέση ενός κινητού χρήστη μπορεί να φανεί σε μια σειρά υπηρεσιών βασισμένων στην πλοήγηση. Για παράδειγμα

με τον προσδιορισμό της θέσης ενός κινητού τηλεφώνου, ο χειριστής μπορεί να ενημερώσει το χρήστη ακριβώς που είναι όπως επίσης και να του δώσει λεπτομερείς κατευθύνσεις για το πώς να φτάσει στον επιθυμητό προορισμό.

1.5.1.2 Υπηρεσίες έρευνας και πληροφοριών

Ο απλούστερος και μέχρι τώρα ο πιο διαδεδομένος τύπος LBS είναι οι *υπηρεσίες έρευνας* και *πληροφοριών*, που παρέχουν στον κινητό χρήστη τα κοντινά *σημεία ενδιαφέροντος (Points of Interest - POI)* όπως εστιατόρια ή τράπεζες. Όταν πραγματοποιηθεί μια αίτηση, ο χρήστης είτε ανιχνεύεται αυτόματα από το κινητό δίκτυο ή, εάν δεν υπάρχει η κατάλληλη τεχνολογία προσδιορισμού θέσης, τότε πρέπει να εισάγει χειροκίνητα την τρέχουσα θέση του. Επιπλέον, πρέπει να διευκρινίσει τα σημεία ενδιαφέροντος, για παράδειγμα, εάν θα επιθυμούσε να λάβει έναν κατάλογο όλων των κοντινών εστιατορίων ή των τραπεζών, και την επιθυμητή μέγιστη απόσταση μεταξύ της τρέχουσας θέσης του και των σημείων ενδιαφέροντος. Το αίτημα έπειτα περνά στον πάροχο των υπηρεσιών, ο οποίος συγκεντρώνει έναν κατάλογο με τα κατάλληλα σημεία ενδιαφέροντος και τα επιστρέφει στο χρήστη. Κατά συνέπεια, αυτός ο τύπος υπηρεσίας είναι βασικά μια επέκταση ενός χρυσού οδηγού για την παρουσίαση μόνο των επιθυμητών τοπικών καταχωρήσεων. Σε μερικές περιπτώσεις αυτές οι υπηρεσίες συνδυάζονται με την παρουσίαση στο χρήστη του συντομότερου μονοπατιού προς τα σημεία ενδιαφέροντος.

1.5.1.3 Community services

Όπως είναι γνωστό τα community services επιτρέπουν σε χρήστες με κοινά ενδιαφέροντα να δημιουργήσουν μια κλειστή ομάδα χρηστών (*community*) για να αλληλεπιδράσουν μεταξύ τους είτε μέσω chat ή των υπηρεσιών messaging. Μια πολύ δημοφιλής υπηρεσία αυτού του τύπου είναι το *Instant Messaging*. Εάν ένας χρήστης είναι εγγραμμένος στην υπηρεσία, μπορεί να παρατηρήσει ποιος από τους φίλους του είναι επίσης συνδεδεμένος και μπορεί να επικοινωνήσει μαζί του.

Τα community services δεν έχουν διαδοθεί ακόμα στα κινητά δίκτυα όμως η αναβάθμισή τους με κάποια χαρακτηριστικά των LBS είναι πιθανό να τα κάνει δημοφιλή και σε αυτά. Χαρακτηριστικές λειτουργίες τους μπορεί να είναι η παρουσίαση στο χρήστη της τρέχουσας θέσης των φίλων του ή να τον προειδοποιήσουν εάν ένας από αυτούς είναι σε κοντινή απόσταση. Τα community services απαιτούν μόνιμη ανίχνευση των μελών της υπηρεσίας και περίπλοκους μηχανισμούς για την διατήρηση της ιδιωτικότητας τους[6].

1.5.1.4 Υποστήριξη των οχημάτων μέσω της τηλεπληροφορικής

Ο τομέας της υποστήριξης των οχημάτων μέσω της τηλεπληροφορικής στοχεύει να υποστηρίξει τους οδηγούς αυτοκινήτων με ένα σύνολο πολλαπλών υπηρεσιών σχετικά με τα οχήματά τους. Περιλαμβάνει την πλοήγηση, την αυτόματη παραμετροποίηση των συσκευών και τα προστιθέμενα χαρακτηριστικά γνωρίσματα μέσα στο όχημα, τα διαγνωστικά των δυσλειτουργιών και τη διάδοση των μηνυμάτων προειδοποίησης, αλλά δεν περιορίζεται σ' αυτά. Η πιο διαδεδομένη εφαρμογή μέχρι τώρα είναι η πλοήγηση, η οποία ενεργοποιείται από τις *On-Board Units* (OBU) που εγκαθίστανται στα αυτοκίνητα. Βάσει της τρέχουσας θέσης, η οποία λαμβάνεται μέσω GPS, το OBU καθοδηγεί τον οδηγό στον επιθυμητό στόχο δίνοντας είτε φωνητικές οδηγίες είτε με την επίδειξη της διαδρομής γραφικά. Οι περιπλοκότερες εκδόσεις αυτών των συστημάτων είναι εξοπλισμένες με τις μονάδες GSM/GPRS και μπορούν έτσι να τηρήσουν τον οδηγό ενήμερο με πληροφορίες από ένα απομακρυσμένο server, όσον αφορά την πιο πρόσφατη κυκλοφοριακή συμφόρηση, τις καιρικές συνθήκες και οδικές εργασίες. Βάσει αυτών των πληροφοριών, είναι δυνατό να δημιουργήσουν και εναλλακτικές διαδρομές.

1.5.1.5 Υπηρεσίες παρακολούθησης και διαχείρισης

Οι υπηρεσίες παρακολούθησης και διαχείρισης μπορούν να ισχύσουν εξίσου και στον καταναλωτή και τις εταιρικές αγορές. Ένα δημοφιλές παράδειγμα αναφέρεται στην παρακολούθηση των ταχυδρομικών συσκευασιών έτσι ώστε οι επιχειρήσεις να ξέρουν που είναι τα προϊόντα τους οποιαδήποτε στιγμή. Η παρακολούθηση οχημάτων μπορεί επίσης να εφαρμοστεί στον εντοπισμό και την αποστολή του κοντινότερου ασθενοφόρου σε μια δεδομένη κλήση. Μια παρόμοια εφαρμογή επιτρέπει στις επιχειρήσεις να εντοπίζουν το προσωπικό τους (πωλητές, μηχανικούς) έτσι ώστε να είναι σε θέση να στείλουν τον κοντινότερο μηχανικό και να παρέχουν στους πελάτες τους ακριβείς χρόνους άφιξης του. Τέλος, είναι δυνατόν να γίνει ακριβής παρακολούθηση των προϊόντων μέσα στην αλυσίδα ανεφοδιασμού.

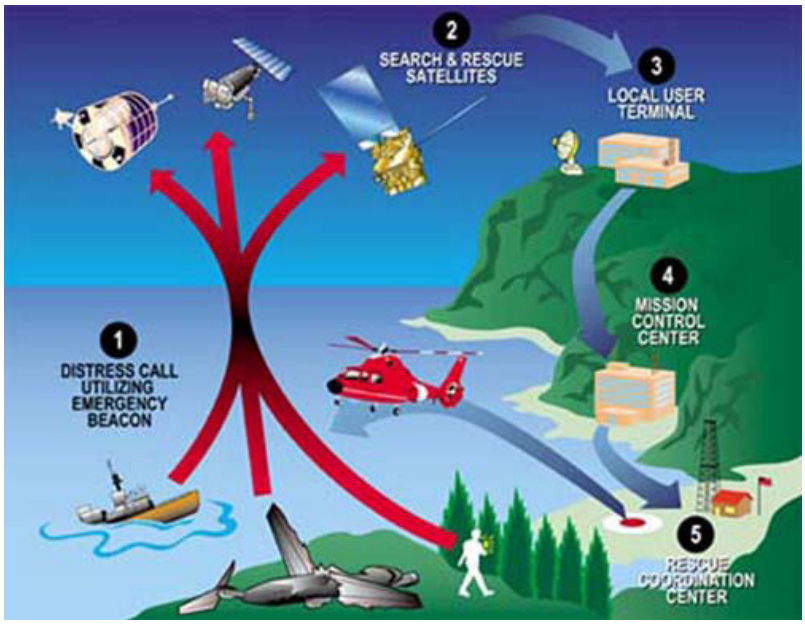
1.5.2. Κρατικές Πρωτοβουλίες

Σε πολλές χώρες του κόσμου, οι κυβερνήσεις και οι αρχές έχουν αναγνωρίσει τις δυνατότητες από τα νέα συστήματα επικοινωνιών και τα χρησιμοποιούν για την υποστήριξη και την εκπλήρωση επιχειρησιακών και διοικητικών στόχων. Προφανώς, οι νέες τεχνικές δυνατότητες που προκύπτουν από τα LBS έχουν ωθήσει πολλές κυβερνήσεις να σκεφτούν τη χρήση των νέων υπηρεσιών για διάφορους εθνικούς λόγους, όπως η αντιμετώπιση της εγκληματικότητας και η συλλογή των φόρων. Όπως

γίνεται κατανοητό οι κρατικές πρωτοβουλίες μπορούν να αποτελέσουν τις κατευθυντήριες δυνάμεις για μια ευρεία εμπορική εισαγωγή των LBS.

1.5.2.1 Υπηρεσίες έκτακτης ανάγκης

Μια από τις προφανέστερες εφαρμογές των LBS είναι η δυνατότητα να εντοπισθεί ένα πρόσωπο που είτε δεν γνωρίζει την ακριβή του θέση ή δεν μπορεί να την αποκαλύψει λόγω μιας κατάστασης έκτακτης ανάγκης (τραυματισμός, εγκληματική επίθεση κλπ). Για παράδειγμα οι οδηγοί αυτοκινήτων συχνά δεν γνωρίζουν την ακριβή τους θέση όταν στο όχημά τους συμβεί βλάβη. Όπως φαίνεται στην **Εικόνα 4** με τη βοήθεια των LBS η ακριβής θέση μεταφέρεται αυτόματα στις υπηρεσίες έκτακτης ανάγκης έτσι ώστε αυτές να είναι σε θέση να παράσχουν τη βοήθεια τους γρήγορα και αποτελεσματικά. Αυτή η κατηγορία περιλαμβάνει τις δημόσιες και ιδιωτικές υπηρεσίες έκτακτης ανάγκης και για τους πεζούς και για τους οδηγούς.



Εικόνα 4: Επισκόπηση υπηρεσιών έκτακτης ανάγκης

1.5.2.2 Συστήματα διοδίων

Σε πολλές χώρες, οι οδηγοί πρέπει να πληρώσουν διόδια για τη διέλευσή τους από εθνικές οδούς, γέφυρες ή και σήραγγες. Υπάρχουν δύο τρόποι για την πληρωμή των διοδίων. Ο πρώτος είναι με την απ' ευθείας πληρωμή στο προσωπικό των σταθμών διοδίων κατά τη διέλευση των οδηγών από αυτούς. Εναλλακτικά οι οδηγοί μπορούν να αγοράσουν κάρτες που ισχύουν για μια ορισμένη χρονική διάρκεια, συνήθως ένα μήνα ή ένα έτος. Ενώ η πρώτη προσέγγιση προκαλεί φρικτές συμφορήσεις στους δρόμους, η κάρτα πάσχει από το γεγονός ότι ένας οδηγός δεν μπορεί να χρεωθεί ανάλογα με την

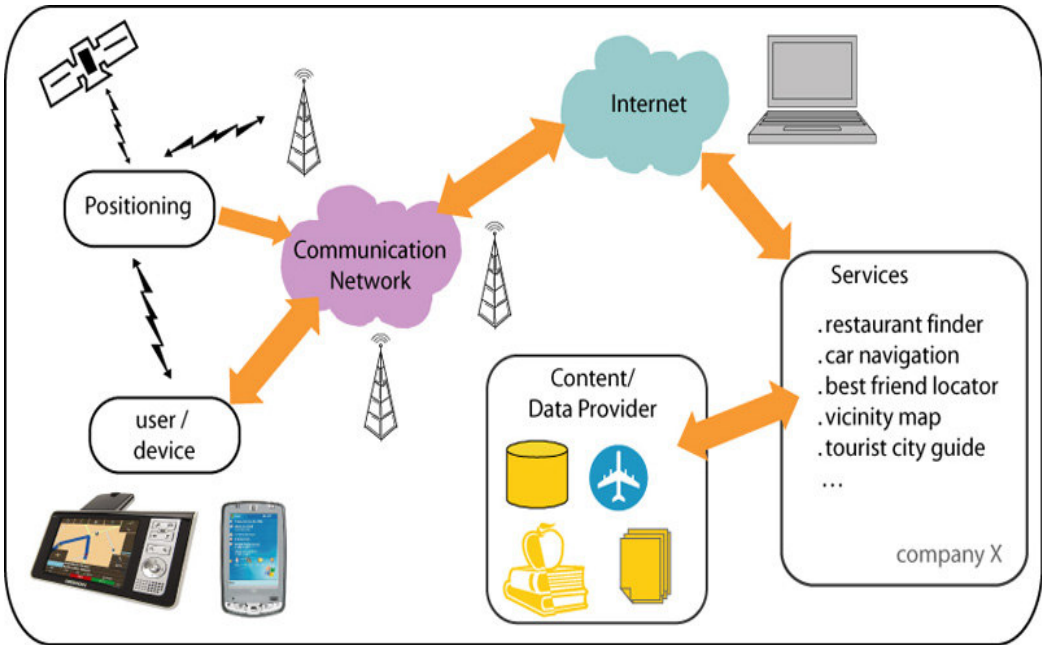
απόσταση που καλύπτει. Για να αντιμετωπίσουν αυτά τα προβλήματα, πολλές χώρες έχουν προωθήσει δραστηριότητες που στοχεύουν στην καταγραφή της χρήσης των δρόμων και της συλλογής των διοδίων ηλεκτρονικά. Μερικά συστήματα που έχουν αναπτυχθεί μέχρι τώρα απαιτούν κάθε όχημα να είναι εξοπλισμένο με ένα OBU, το οποίο ανταλλάσσει τα στοιχεία ασύρματα με σταθερούς σταθμούς ελέγχου που βρίσκονται κατά μήκος των δρόμων. Η επεξεργασία αυτών των στοιχείων από το OBU ανιχνεύει τη χρήση των εθνικών οδών και καθορίζει την απόσταση που έχει καλυφθεί. Το πρόβλημα μέχρι τώρα είναι ότι οι Ευρωπαϊκές χώρες δεν έχουν υιοθετήσει κάποιο πρότυπο και τα συστήματα σε διαφορετικές χώρες δεν είναι συμβατά[5].

ΚΕΦΑΛΑΙΟ 2

ΑΡΧΙΤΕΚΤΟΝΙΚΗ LBS ΣΥΣΤΗΜΑΤΩΝ

2.1. Επεξεργασία αιτήματος εξυπηρέτησης LBS

Σ' αυτή την ενότητα θα αναλύσουμε τη διαδικασία που ακολουθείται, στο παράδειγμα του κινέζικου εστιατορίου, από τη στιγμή της δημιουργίας ενός αιτήματος εξυπηρέτησης μέχρι την απάντηση που θα δοθεί. Η όλη διαδικασία φαίνεται στην **Εικόνα 5**. Η πληροφορία που θέλει ο χρήστης είναι η διαδρομή σε ένα κοντινό κινέζικο εστιατόριο. Επομένως ο χρήστης εκφράζει την ανάγκη του επιλέγοντας την κατάλληλη λειτουργία στην κινητή συσκευή του.



Εικόνα 5: Τμήματα των LBS συστημάτων και ροή της πληροφορίας

Αν η λειτουργία έχει ενεργοποιηθεί, η πραγματική θέση της κινητής συσκευής λαμβάνεται από την Υπηρεσία Προσδιορισμού Θέσης. Αυτό μπορεί να γίνει είτε από την ίδια τη συσκευή χρησιμοποιώντας GPS είτε με μια δικτυακή υπηρεσία προσδιορισμού θέσης. Κατόπιν ο κινητός πελάτης στέλνει το αίτημα με τις πληροφορίες που θέλει και τη θέση του αντικειμένου που ζητάει, μέσω του δικτύου επικοινωνίας σε ένα gateway.

Το gateway έχει την αποστολή να ανταλλάξει τα μηνύματα μεταξύ του κινητού δικτύου επικοινωνίας και του Internet. Επομένως ξέρει τις web διευθύνσεις διαφόρων εξυπηρετητών και καθοδηγεί το αίτημα σε έναν τέτοιο εξυπηρετητή. Το gateway

αποθηκεύει επίσης τις πληροφορίες για την κινητή συσκευή που έχει ζητήσει τις πληροφορίες.

Ο application server διαβάζει το αίτημα και ενεργοποιεί την αρμόδια υπηρεσία - στην περίπτωσή μας μια χωρική υπηρεσία αναζήτησης.

Τώρα, το LBS σύστημα αναλύει ξανά το μήνυμα και αποφασίζει ποιες πρόσθετες πληροφορίες εκτός από τα κριτήρια αναζήτησης (εστιατόριο + κινέζικο) και τη θέση των χρηστών απαιτούνται για να δοθεί απάντηση στο αίτημα. Στην περίπτωσή μας το σύστημα θα διαπιστώσει ότι χρειάζεται τις πληροφορίες για τα εστιατόρια από το χρυσό οδηγό μιας συγκεκριμένης περιοχής και επομένως θα ζητήσει αυτά τα στοιχεία από έναν πάροχο δεδομένων.

Επιπλέον το σύστημα θα βρει τις πληροφορίες που απαιτούνται για τους δρόμους για να ελέγξει εάν υπάρχει δυνατότητα πρόσβασης στο εστιατόριο (π.χ. μερικές φορές ένα εστιατόριο από την άλλη πλευρά ποταμών μπορεί να μην είναι προσπελάσιμο αν δεν υπάρχει κοντά γέφυρα).

Έχοντας τώρα όλες τις πληροφορίες το σύστημα θα κάνει μια χωρική ερώτηση δρομολόγησης για να πάρει σαν αποτέλεσμα μερικά κινέζικα εστιατόρια. Μετά την εύρεση μιας λίστας με τα κοντινά εστιατόρια το αποτέλεσμα στέλνεται πίσω στο χρήστη μέσω του Internet, του gateway και του κινητού δικτύου.

Τα εστιατόρια θα παρουσιαστούν τώρα στο χρήστη είτε ως κατάλογος κειμένου (π.χ. ταξινομημένα με αύξουσα σειρά βάσει της απόστασης από το σημείο που βρίσκεται ο χρήστης) ή σχεδιασμένα σε ένα χάρτη. Κατόπιν ο χρήστης θα μπορούσε να ζητήσει περισσότερες πληροφορίες για τα εστιατόρια (π.χ. το μενού και τις τιμές), πράγμα το οποίο ενεργοποιεί ένα διαφορετικό είδος υπηρεσιών. Τέλος εάν έχει επιλέξει ένα συγκεκριμένο εστιατόριο μπορεί να ζητήσει τη διαδρομή γι' αυτό το εστιατόριο.

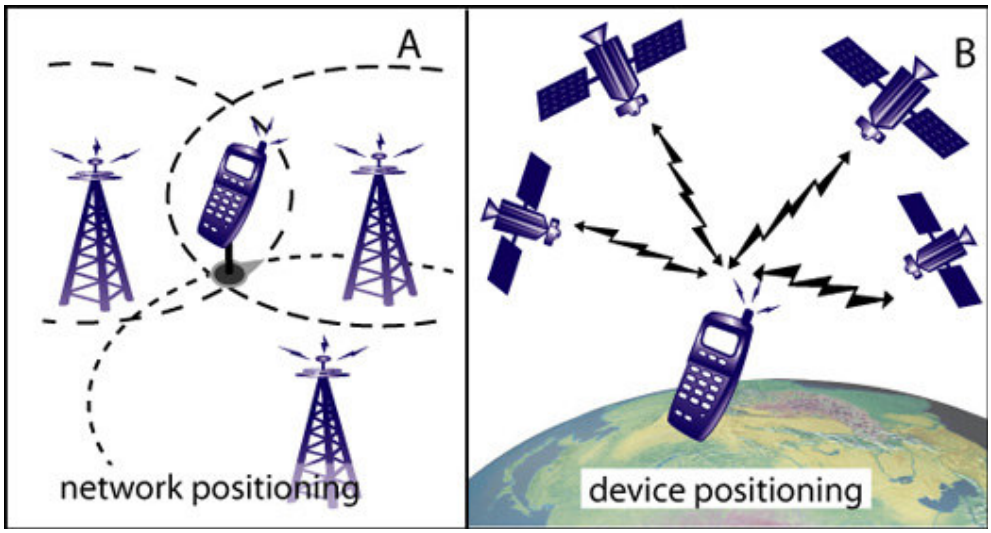
2.2. Μέθοδοι προσδιορισμού θέσης

Στην προηγούμενη ενότητα αναφερθήκαμε στο σύστημα προσδιορισμού θέσης, το οποίο λαμβάνει τη θέση των χρηστών. Σ' αυτή την ενότητα θα αναλύσουμε τις μεθόδους προσδιορισμού θέσης. Εάν δεν θεωρούμε τη χειροκίνητη εισαγωγή της θέσης ως μέθοδο προσδιορισμού θέσης, τότε μια γενική ταξινόμηση των μεθόδων προσδιορισμού θέσης είναι με το διαχωρισμό τους σε δύο ομάδες: Η πρώτη ομάδα προσδιορισμού θέσης είναι βασισμένη στο δίκτυο (network based positioning). Εδώ η αναζήτηση και η αξιολόγηση της θέσης των χρηστών γίνεται με τη χρησιμοποίηση ενός δικτύου σταθμών βάσεων (βλ.Εικόνα 6: **Μέθοδοι προσδιορισμού θέσης σε εξωτερικούς χώρους**).

Επομένως η κινητή συσκευή είτε στέλνει ένα σήμα είτε ανιχνεύεται από το δίκτυο. Η δεύτερη ομάδα είναι βασισμένη στο τερματικό (terminal-based positioning). Εδώ, η θέση υπολογίζεται από τα σήματα που λαμβάνει η συσκευή του χρήστη από τους σταθμούς βάσεων. Η πιο γνωστή τεχνολογία για ένα βασισμένο στο τερματικό σύστημα είναι η χρήση του GPS. Οι σταθμοί βάσεων αυτού είναι οι GPS δορυφόροι (βλ. το σχήμα B στην Εικόνα 6). Τέλος μια τρίτη ομάδα τεχνικών προσδιορισμού θέσης προκύπτει από το συνδυασμό των δύο παραπάνω τεχνικών.

Οι βασικές αρχές για τον υπολογισμό της θέσης των χρηστών είναι:

- Οι σταθμοί βάσεων έχουν μια γνωστή θέση.
- Η πληροφορία από ένα σήμα μετασχηματίζεται σε αποστάσεις.
- Ο υπολογισμός της θέσης γίνεται χρησιμοποιώντας τις ληφθείσες αποστάσεις στους σταθμούς βάσεων.



Εικόνα 6: Μέθοδοι προσδιορισμού θέσης σε εξωτερικούς χώρους

Οι δύο τεχνολογίες θέσης που κυριαρχούν σήμερα είναι το ήδη αναφερθέν GPS και η αξιολόγηση της θέσης χρησιμοποιώντας τη διεύθυνση κυψέλης κινητής τηλεφωνίας (Cell-ID) η οποία λαμβάνεται από τον κοντινότερο σταθμό βάσης. Το GPS παραδίδει μια πολύ ακριβή θέση (ακρίβεια μέχρι 5m) ενώ το Cell-ID παραδίδει μια πιο προσεγγιστική θέση (ακρίβεια από 100m μέχρι 1χλμ). Ειδικά το GPS είναι (αυτήν την περίοδο) μια outdoor μέθοδος προσδιορισμού θέσης. Για να λάβουμε εσωτερικές θέσεις με υψηλή ακρίβεια, όπως απαιτείται παραδείγματος χάριν στα μουσεία ή στα μεγάλα εμπορικά κέντρα, μέθοδοι εντοπισμού θέσης βασισμένες σε WLAN, Bluetooth ή

υπέρυθρες τεχνολογίες πρέπει να εφαρμοστούν. Οι συγκεκριμένες τεχνολογίες για τον προσδιορισμό της θέσης σε εσωτερικούς χώρους θα αναλυθούν πιο διεξοδικά στην επόμενη ενότητα. Γενικά είναι σημαντικό να σημειωθεί ότι η τεχνολογία θέσης και η ακρίβειά της επηρεάζουν την εφαρμογή των διαφορετικών υπηρεσίες θέσης.

2.3. Πρότυπα υπηρεσιών προσδιορισμού θέσης

Στις προηγούμενες ενότητες συζητήσαμε τις απαιτήσεις και τα απαραίτητα τμήματα των LBS συστημάτων. Όπως είδαμε σε ένα LBS σύστημα εμπλέκονται διάφοροι φορείς που κυμαίνονται από τους παρόχους τεχνολογίας ως τους παρόχους δεδομένων. Για να διασφαλιστεί ότι όλες οι διαφορετικές τεχνολογίες και συσκευές μπορούν να λειτουργήσουν μαζί πρέπει να καθοριστούν κοινά πρότυπα για τις διεπαφές των επιμέρους στοιχείων. Οι πλέον σημαντικοί φορείς έκδοσης προτύπων στο χώρο των υπηρεσιών παροχής πληροφορίας θέσης είναι ο Open Geospatial Consortium, Inc (OGC), η ομάδα εργασίας Open Mobile Alliance's Location Working Group (LOC) και το International Standard Organization (ISO).

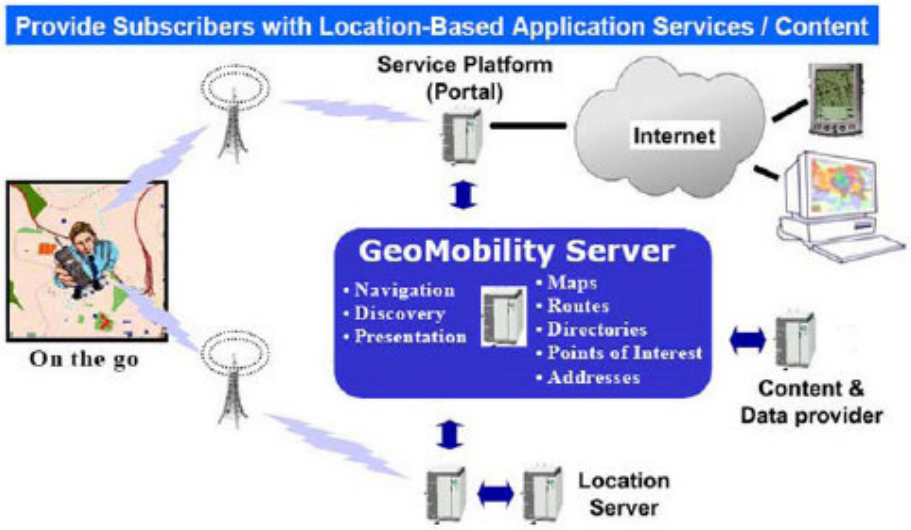
Το OGC το οποίο ιδρύθηκε το 1994 είναι μια διεθνής συνεργασία όπου συμμετέχουν εταιρείες, ακαδημαϊκοί φορείς και κυβερνητικοί οργανισμοί και η οποία εστιάζεται στην ανάπτυξη web προτύπων στην ευρεία geospatial αγορά που περιλαμβάνει γεωγραφικά πληροφοριακά συστήματα (GIS) καθώς και υπηρεσίες χαρτογράφησης, απεικόνισης της γης, κινητών και ασυρμάτων τεχνολογιών. Μεταξύ άλλων τα πρότυπα που έχουν αναπτυχθεί από το OGC είναι τα OpenLS, WFS, WMS, GML και SLD.

Το OGC προσφέρει επίσης ένα πρόγραμμα ελέγχου συμβατότητας το οποίο έχει σαν στόχο να βοηθήσει τόσο τις εταιρείες όσο και τους χρήστες να εκμεταλλευτούν τα πλεονεκτήματα που προσφέρει η υλοποίηση των προδιαγραφών του OpenGIS. Το πρόγραμμα παρέχει μια διαδικασία για τον έλεγχο συμβατότητας των προϊόντων του OpenGIS Implementation Specifications καθώς και για τη διαλειτουργικότητα μεταξύ συμβατών προϊόντων.

Το τελευταίο πεδίο ενδιαφέροντος του OGC είναι το sensor web. Αναπτύσσεται μια ανοικτή πλατφόρμα για την αξιοποίηση αισθητήρων που είναι συνδεδεμένοι μέσω διαδικτύου. Η εργασία αυτή περιλαμβάνει το SensorML το οποίο είναι ένα πληροφοριακό μοντέλο που κάνει χρήση XML για την εύρεση, αναζήτηση και έλεγχο αισθητήρων που βρίσκονται στο διαδίκτυο. Στη συνέχεια περιγράφονται τα κυριότερα πρότυπα των υπηρεσιών προσδιορισμού θέσης.

2.3.1. OpenLS

Λαμβάνοντας υπόψη το ISO 19119, το οποίο παρέχει ένα γενικό πλαίσιο της υπηρεσίας και το ISO 19101, που δίνει μια ταξινόμηση των γεωγραφικών υπηρεσιών, το OGC εξέδωσε τις προδιαγραφές για τα Open Location Services (OpenLS - Open Geospatial Consortium 2005) [1]. Το OpenLS καθορίζει τις κύριες υπηρεσίες, την πρόσβαση και τους τύπους δεδομένων που διαμορφώνουν μαζί ένα πλαίσιο για μια ανοικτή πλατφόρμα υπηρεσιών, που ονομάζεται *GeoMobility server*. Ο server αυτός λειτουργεί σαν application server και πρέπει να απαντάει και να προωθεί κεντρικά αιτήματα εξυπηρέτησης. Ο ρόλος αυτού του server απεικονίζεται στην Εικόνα 7. Πρέπει να παρατηρηθεί ότι τα αιτήματα εξυπηρέτησης σε ένα GeoMobility server μπορούν να σταλούν από ένα κινητό χρήστη, από χρήστες του Internet αλλά και από άλλους application servers.



Εικόνα 7: Ο ρόλος του GeoMobility server

Οι κύριες υπηρεσίες που ορίζονται στις προδιαγραφές του OpenLS 1.1 [1] περιλαμβάνουν πέντε τύπους υπηρεσίας:

Directory Service (χωρικός χρυσός οδηγός): Αυτή η υπηρεσία παρέχει στους συνδρομητές πρόσβαση σε ένα online directory για να βρουν την κοντινότερη ή τη συγκεκριμένη θέση, το προϊόν ή την υπηρεσία.

Παράδειγμα 1: "Πού είναι το κινέζικο εστιατόριο Golden Phoenix;"

Παράδειγμα 2: "Πού υπάρχουν κινέζικα εστιατόρια;"

Gateway service: Αυτή η υπηρεσία είναι η διεπαφή μεταξύ του GeoMobility Server και του Location Server από την υπηρεσία προσδιορισμού θέσης (βλ. Εικόνα 7). Είναι χρήσιμο το αίτημα για την τρέχουσα θέση να γίνεται με διαφορετικούς τρόπους (π.χ. ένα ή πολλά τερματικά, άμεση ή περιοδική θέση).

Location Utility Service (Geocode/ Reverse Geocode): Αυτή η υπηρεσία αποδίδει ένα Geocode, καθορίζοντας μια γεωγραφική θέση, εάν ένα όνομα περιοχής, διεύθυνση οδών ή ο ταχυδρομικός κώδικας δίνεται. Αποδίδει επίσης ένα αντίστροφο Geocode με τον καθορισμό ενός πλήρους ονόματος θέσης/διεύθυνσης, οδών/ταχυδρομικός κώδικας, για μια δεδομένη γεωγραφική θέση.

Παράδειγμα 1: Δοθείσης μιας διεύθυνσης, βρείτε μια θέση.

Παράδειγμα 2: Οδήγησε σε μια διεύθυνση (θέση).

Παράδειγμα 3: Δοθείσης μια θέσης, βρείτε μια διεύθυνση.

Παράδειγμα 4: "Πού είμαι;"

Presentation Service: Αυτή η υπηρεσία δίνει τις γεωγραφικές πληροφορίες για την παρουσίαση σε ένα κινητό τερματικό. Μια εφαρμογή OpenLS μπορεί να ζητήσει από αυτήν την υπηρεσία να λάβει ένα χάρτη μιας επιθυμητής περιοχής, με ή χωρίς επικαλύψεις χαρτών που απεικονίζουν τη γεωμετρία των διαδρομών, τα σημεία ενδιαφέροντος, την περιοχή ενδιαφέροντος, την τοποθεσία, τη θέση και τη διεύθυνση.

Route Service: Αυτή η υπηρεσία καθορίζει μια διαδρομή για έναν συνδρομητή. Ο χρήστης πρέπει να καθορίσει το σημείο έναρξης (συνήθως η θέση που αποκτιέται μέσω του Gateway Service, αλλά αυτό θα μπορούσε επίσης να είναι μια καθορισμένη θέση, π.χ. το σπίτι τους για ένα προγραμματισμένο ταξίδι), και το σημείο τέλους (οποιαδήποτε θέση, όπως μια θέση για την οποία έχουν μόνο τον τηλεφωνικό αριθμό ή μια διεύθυνση, ή μια θέση που αποκτιέται μέσω μιας αναζήτησης σε ένα Directory Service). Ο συνδρομητής μπορεί προαιρετικά να διευκρινίσει κάποια σημεία από τα οποία θα ήθελε να περάσει, με κάποιο τρόπο, ή την προτίμηση διαδρομών (γρηγορότερη, κοντινότερη, αυτή με τη λιγότερη κυκλοφορία, κ.λπ.), και τον προτιμημένο τρόπο μετάβασης. Οι επιστρεφόμενες πληροφορίες δρομολόγησης μπορεί να είναι σε μορφή κειμένου, σε έναν κώδικα παρουσίασης (που περιγράφει στροφές και αποστάσεις) ή γεωμετρικές, οι οποίες είναι χρήσιμες για έναν χάρτη.

2.3.2. Web Feature Service

Η υπηρεσία Web Feature (WFS) προσδιορίζει τις διεπαφές για την περιγραφή των λειτουργιών διαχείρισης δεδομένων σε γεωγραφικά γνωρίσματα (features) κάνοντας χρήση της HTTP. Οι λειτουργίες διαχείρισης των δεδομένων περιλαμβάνουν τόσο αναζητήσεις βασισμένες σε χωρικές και μη χωρικές παραμέτρους (κυρίως WFS) όσο και τη δυνατότητα δημιουργίας, ενημέρωσης και διαγραφής feature instances (WFS with transactions).

Ένα WFS αίτημα αποτελείται από την περιγραφή λειτουργιών αναζήτησης ή μετασχηματισμού δεδομένων που θα εφαρμοστούν σε ένα ή περισσότερα γνωρίσματα (features). Το αίτημα δημιουργείται στον client και μεταβιβάζεται στον web feature server μέσω http. Ο web feature server διαβάζει και εκτελεί το αίτημα. Η WFS επιστρέφει στον client δεδομένα κωδικοποιημένα σε Geography Markup Language (GML), ο οποίος τελικά αποφασίζει πως θα τα επεξεργαστεί περαιτέρω.

2.3.3. Web Map Service

Η WMS προδιαγραφή τυποποιεί τον τρόπο που οι χάρτες ζητούνται από τον client καθώς και τον τρόπο που ο server θα περιγράφει τα δεδομένα που διαθέτει. Η WMS ορίζει τρεις λειτουργίες:

- GetCapabilities (απαραίτητη): Εξασφάλιση μεταδεδομένων επιπέδου υπηρεσίας τα οποία αποτελούν μια περιγραφή του περιεχομένου της πληροφορίας της υπηρεσίας και των επιτρεπτών παραμέτρων των αιτημάτων
- GetMap (απαραίτητη): Εξασφάλιση μιας εικόνας χάρτη του οποίου οι γεωγραφικές και διαστατικές παράμετροι είναι καλώς ορισμένες
- GetFeatureInfo (προαιρετική): Ερώτημα για παροχή πληροφορίας σχετικά με συγκεκριμένα γνωρίσματα (features) που εμφανίζονται στο χάρτη

Κατά την υποβολή αιτήματος για παροχή χάρτη, ο client μπορεί να προσδιορίσει την πληροφορία που θα εμφανίζεται στο χάρτη (ένα ή περισσότερα layer), πιθανόν το στυλ των layers, το τμήμα της γης που θα απεικονιστεί (περιβάλλον box), το σύστημα συντεταγμένων που θα χρησιμοποιηθεί (SRS), το επιθυμητό τύπο εικόνας με το οποίο θα απεικονιστεί ο χάρτης, το μέγεθος (πλάτος και μήκος) της εικόνας, το χρώμα και τη διαφάνεια (transparency) της εικόνας. Δηλαδή το WMS πρωτόκολλο ορίζει μια απλή διεπαφή για web based εφαρμογές απεικόνισης χαρτών το οποίο βασίζεται σε απλό συντακτικό για την υποβολή αιτήματος και τη λήψη απόκρισης για ένα χάρτη ο οποίος θα απεικονιστεί με τυποποιημένο είδος γραφικού (GIF, PNG, SVG κλπ).

Το WMS επιτρέπει μόνο στοιχειώδεις στιλιστικές μορφές απεικόνισης των γεωγραφικών δεδομένων. Για το λόγο αυτό αναπτύχθηκε η επέκταση Styled Layer Desriptor (SLD). Η SLD είναι μια κωδικοποίηση για το πώς η προδιαγραφή του WMS μπορεί να επεκταθεί για να επιτρέψει συμβολισμό των δεδομένων οριζόμενων από το χρήστη το οποίο σημαίνει ότι επιτρέπεται η παραγωγή περισσότερο χρηστικών χαρτών.

2.3.4. Geography Markup Language (GML)

Η GML είναι μια XML κωδικοποίηση για τη μοντελοποίηση, μεταφορά και αποθήκευση γεωγραφικής πληροφορίας. Η προδιαγραφή GML ορίζει ένα XML συντακτικό για την περιγραφή της γραμματικής στην οποία θα συμμορφώνονται τα GML δεδομένα. Η GML είναι αρθρωτής αρχιτεκτονικής που σημαίνει ότι μπορούμε να διαλέγουμε το schema εκείνο που κάθε φορά ανταποκρίνεται στις ανάγκες μας.

Η GML παρέχει διαφορετικά είδη αντικειμένων για την περιγραφή γεωγραφικών δεδομένων όπως σύστημα συντεταγμένων, χρόνος, τοπολογία, γεωμετρία και γνωρίσματα. Ένα γεωγραφικό γνώρισμα είναι η αφηρημένη έννοια ενός χαρακτηριστικού του πραγματικού κόσμου. Επομένως η ψηφιακή αναπαράσταση του πραγματικού κόσμου μπορεί να θεωρηθεί ότι είναι ένα σύνολο γνωρισμάτων.

Στα πλεονεκτήματα της χρήσης GML περιλαμβάνεται η διαλειτουργικότητά της (υποστήριξη από πολλές εταιρείες), το ότι στηρίζεται σε XML, η ευελιξία οπτικοποίησης των δεδομένων και η δυνατότητα χρήσης ευέλικτων και αποτελεσματικών ερωτημάτων. Το μειονέκτημα της χρήσης GML είναι ότι το μέγεθος των αρχείων που δημιουργούνται είναι μεγαλύτερο από τα αντίστοιχα αρχεία που έχουν μορφή binary όπως για παράδειγμα τα ESRI Shapefiles.

2.3.5. OMA Location Working Group

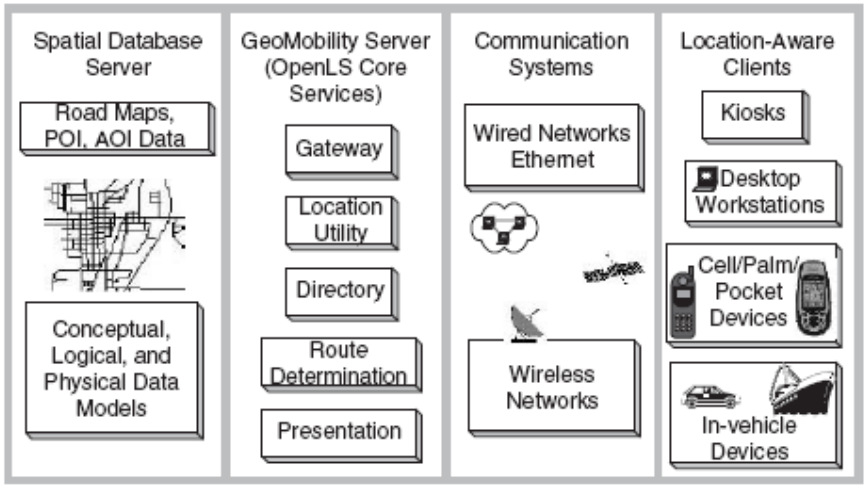
Η ομάδα εργασίας Open Mobile Alliance's Location Working Group (LOC), η οποία παλαιότερα ήταν γνώστη ως Location Interoperability Form, αναπτύσσει προδιαγραφές για τη διασφάλιση της διαλειτουργικότητας των Mobile Location Services. Το πλέον γνωστό αποτέλεσμα της εργασίας της OMA LOC είναι το Mobile Location Protocol (MLP).

2.4. Spatial Databases and GIS

Όπως αναφέρθηκε και στις προηγούμενες ενότητες τα LBS συστήματα παρέχουν τη δυνατότητα εύρεσης της γεωγραφικής θέσης μιας κινητής συσκευής και τότε παρέχουν υπηρεσίες βασισμένες σ' αυτή την θέση (OpenLS). Αυτές οι βασισμένες στη θέση υπηρεσίες απαιτούν ένα Spatial Database (SDB) server, ο οποίος παρέχει την

αποτελεσματική και αποδοτική ανάκτηση και διαχείριση των geospatial δεδομένων. Τα χωρικά συστήματα βάσεων δεδομένων παραδίδουν διάφορα χωρικά δεδομένα (π.χ. ψηφιακοί οδικοί χάρτες) και μη χωρικές πληροφορίες (π.χ., οδηγίες καθοδήγησης διαδρομών) στον πελάτη μετά από αίτησή του. Οι SDB servers παρέχουν αποδοτικές geospatial ικανότητες επεξεργασίας ερωτημάτων όπως η εύρεση του κοντινότερου σε μια συγκεκριμένη θέση βενζινάδικου και εύρεσης της κοντινότερης πορείας στον προορισμό. Το SDB σύστημα ενεργεί ως back-end server στον GeoMobility server. Κατά συνέπεια οι SDB servers διαδραματίζουν έναν κρίσιμο ρόλο στην υλοποίηση αποδοτικών και περίπλοκων εφαρμογών συστημάτων πλοήγησης.

Η γενική αρχιτεκτονική ενός σύγχρονου συστήματος πλοήγησης παρουσιάζεται στην Εικόνα 8. Τα τμήματα μπορούν να ταξινομηθούν ευρέως σε τέσσερα υποσυστήματα: στον SDB server, στον GeoMobility server, στα συστήματα επικοινωνίας, και στους location-aware clients.



Εικόνα 8: Αρχιτεκτονική ενός σύγχρονου συστήματος πλοήγησης

2.4.1. Ορισμοί DBMS και GIS

Όπως γίνεται κατανοητό οι χωρικές βάσεις δεδομένων και τα Γεωγραφικά Συστήματα Πληροφοριών (GIS) είναι οι βασικές τεχνολογίες για την πραγματοποίηση εργασιών όπως η εύρεση σημείων ενδιαφέροντος και η παροχή υπηρεσιών για τα σημεία αυτά. Χρησιμοποιούνται για να μετατρέπουν την χωρική θέση σε κατανοητές περιγραφικές πληροφορίες για τη θέση και αντίστροφα, που αναφέρονται σαν geocoding και reverse geocoding αντίστοιχα, όπως επίσης και για να δημιουργούν ψηφιακούς χάρτες και πληροφορίες δρομολόγησης, ή για την εύρεση κοντινών σημείων ενδιαφέροντος.

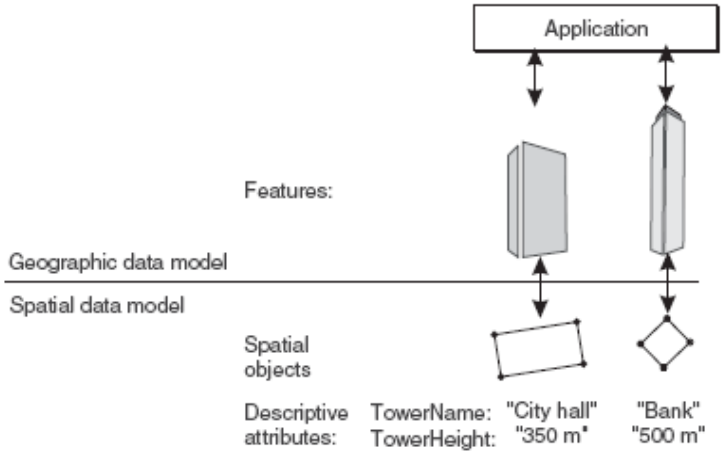
Γενικά, μια βάση δεδομένων δημιουργείται, οργανώνεται, και διατηρείται από ένα *Σύστημα Διαχείρισης Βάσεων Δεδομένων (Database Management System - DBMS*),

που είναι ένα λογισμικό που εκτελείται σ' έναν server για την πραγματοποίηση τέτοιων εργασιών. Σε αυτή τη λογική, μια χωρική DBMS εστιάζει στην αποδοτική αποθήκευση και τη βελτιστοποίηση των χωρικών δεδομένων. Από την άλλη μεριά τα GIS παρερμηνεύονται σαν ένα γεωγραφικό εργαλείο για την παραγωγή χαρτών. Ένας πληρέστερος και περιεκτικότερος ορισμός, ο οποίος χαρακτηρίζει τις λειτουργίες ενός GIS ως εξής: Αποθηκεύει τα γεωγραφικά στοιχεία, ανακτά και συνδυάζει αυτά τα δεδομένα για να δημιουργήσει τις νέες αναπαραστάσεις του γεωγραφικού χώρου, παρέχει τα εργαλεία για χωρική ανάλυση, και εκτελεί τις προσομοιώσεις για να βοηθήσει τους προχωρημένους χρήστες να οργανώσουν την εργασία τους σε πολλές περιοχές, συμπεριλαμβανομένης της δημόσιας διοίκησης, των δικτύων μεταφορών, τις στρατιωτικές εφαρμογές, και περιβαλλοντικά συστήματα πληροφοριών. Ένας άλλος ορισμός που περιγράφει ένα GIS είναι ο εξής: "σύστημα ηλεκτρονικών υπολογιστών για τη συλλογή, τη διαχείριση, την ενσωμάτωση, το χειρισμό, την ανάλυση, και την επίδειξη των δεδομένων που αναφέρονται στη γη."

Σε αυτό το πλαίσιο, είναι σημαντικό να σημειωθεί ότι μια χωρική DBMS αποτελεί ένα αναπόσπαστο τμήμα ενός GIS, που χρησιμοποιείται για τη διατήρηση των χωρικών δεδομένων, και ότι ένα GIS προσφέρει αρκετά διαφορετικές λειτουργίες από μια χωρική DBMS. Στην αγορά, υπάρχουν αρκετά προϊόντα GIS διαθέσιμα, όπως το ArcInfo και το ArcView (και τα δύο από την ESRI). Κάποια από αυτά έχουν ενσωματωμένη μια χωρική DBMS, η οποία συχνά μπορεί να αντικατασταθεί από άλλα συστήματα ενώ υπάρχουν περιπτώσεις όπου οι χωρικές βάσεις είναι διαθέσιμες ως επεκτάσεις των σχεσιακών DBMS (για παράδειγμα η postgis στην postgreSQL).

Για την κατανόηση των GIS, είναι απαραίτητο να γίνει διάκριση μεταξύ δύο αφαιρετικών επιπέδων, τα οποία απεικονίζονται στην Εικόνα 9. Το ανώτερο στρώμα σε ένα GIS, το αποκαλούμενο *γεωγραφικό μοντέλο δεδομένων*, παρέχει μια εννοιολογική άποψη του γεωγραφικού περιεχομένου από την άποψη μονάδων, οι οποίες αποκαλούνται *features*. Ένα feature αντιπροσωπεύει μια πραγματική οντότητα, για παράδειγμα, ένα κτήριο, ένα δρόμο, ένα ποτάμι, μια χώρα, ή μια πόλη. Αποτελείται από ένα *χωρικό στοιχείο*, το οποίο καθορίζει τη θέση του, τη μορφή του και την τοπολογική σχέση του με άλλες οντότητες, και μια *περιγραφή*, η οποία παρέχει μη χωρικές πληροφορίες για την οντότητα, για παράδειγμα, το όνομα μιας πόλης ή ενός δρόμου, ή του πληθυσμού μιας χώρας. Κάθε feature έχει ένα καθορισμένο με σαφήνεια σύνολο διαδικασιών, το οποίο προσαρμόζεται στον τύπο που η πραγματική οντότητα αντιπροσωπεύει. Παραδείγματος χάριν, ένα feature που αντιπροσωπεύει ένα δρόμο μπορεί να παρέχει

μια λειτουργία για να ζητήσει το μήκος του, ενώ ένα feature για μια πόλη μπορεί να προσφέρει μια λειτουργία για τη λήψη του μεγέθους της περιοχής. Οι διαδικασίες χρησιμοποιούνται από τις εφαρμογές για το χειρισμό ή την αναζήτηση της περιγραφικής πληροφορίας ενός feature και της συσχέτισης του με άλλα. Κατά συνέπεια, το γεωγραφικό μοντέλο δεδομένων αντιπροσωπεύει τη διεπαφή μεταξύ του GIS και της εφαρμογής.



Εικόνα 9: Προσέγγιση των layers σε ένα GIS

Το χαμηλότερο layer εξετάζει όλες τις πτυχές της φυσικής διαχείρισης δεδομένων, συμπεριλαμβανομένης της αποθήκευσης, της επεξεργασίας του αιτήματος, και της βελτιστοποίησης, καθώς επίσης και της αποκατάστασης. Ένας ιδιαίτερο ζήτημα όσον αφορά τα GIS είναι η αναπαράσταση των χωρικών τμημάτων των features, τα οποία καλούνται *χωρικά αντικείμενα (spatial objects)*, καθώς επίσης και ο συνδυασμός τους με περιγραφικές ιδιότητες (*descriptive attributes)* για τη διατήρηση των περιγραφών τους. Τα χωρικά αντικείμενα μπορούν να αναπαρασταθούν με πολλούς τρόπους. Γενικά, οι τρόποι αναπαράστασης είναι δύο: *raster mode* και *vector mode*. Αυτοί οι τρόποι αναπαράστασης παρουσιάζονται αναλυτικά στο Κεφάλαιο 4. Στα πλαίσια των GIS, το χαμηλότερο layer που αντιμετωπίζει αυτά τα ζητήματα αναφέρεται ως *μοντέλο χωρικών δεδομένων*.

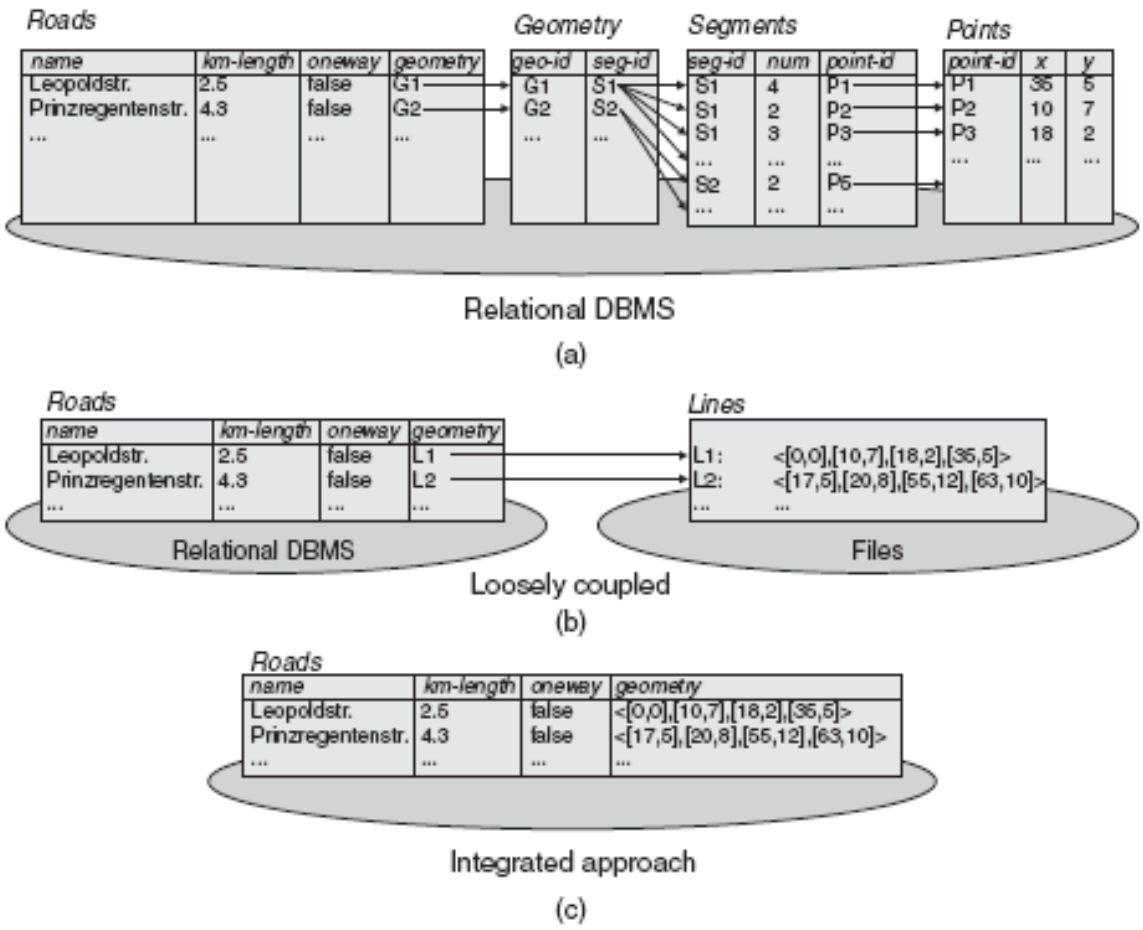
2.4.2. Κατηγορίες βάσεων δεδομένων για χωρικά αντικείμενα

Γενικά, υπάρχει μια διάκριση μεταξύ *σχεσιακών (relational)* και *αντικειμενοστραφών (object-oriented) DBMS* για την αποθήκευση συμβατικών, μη χωρικών δεδομένων. Σε μια σχεσιακή βάση δεδομένων, τα δεδομένα οργανώνονται σε πίνακες όπου κάθε σειρά ενός πίνακα αντιπροσωπεύει μια εγγραφή (record) και κάθε στήλη αντιπροσωπεύει ένα γνώρισμα (attribute). Μια εγγραφή πρέπει να είναι καλά ορισμένη δεδομένου ότι πρέπει

να περιέχει έναν σταθερό αριθμό attributes, και κάθε attribute πρέπει να είναι ενός ορισμένου, απλού τύπου δεδομένων όπως *integer, float, character ή string.* Αυτό είναι δυνατόν να συσχετίζει εγγραφές διαφορετικών πινάκων έτσι ώστε κάθε εγγραφή ενός πίνακα προσδιορίζεται από ένα μοναδικό κλειδί που διαμορφώνει ένα attribute της εγγραφής, και για να χρησιμοποιήσει αυτό το κλειδί ως αναφορά μέσα σε έναν άλλο πίνακα.

Οι αντικειμενοστραφείς DBMS, από την άλλη μεριά, υιοθετούν τους βασικούς μηχανισμούς του αντικειμενοστραφούς μοντέλου, όπως οι κλάσεις, οι μέθοδοι, η ενθυλάκωση (encapsulation), και η κληρονομικότητα για την οργάνωση των δεδομένων. Το αντίστοιχο μιας εγγραφής είναι ένα αντικείμενο που τοποθετεί τα δεδομένα των απλών ή σύνθετων τύπων και παρέχει διάφορες μεθόδους για τον χειρισμό τους. Οι αντικειμενοστραφείς DBMS παρέχουν πολύ καλύτερη απόδοση από τις σχεσιακές, αλλά χρειάζεται περισσότερος χρόνος και δεξιότητες για να σχεδιαστούν. Επομένως μερικά DBMS προϊόντα είναι βασισμένα σε μια υβριδική προσέγγιση που υιοθετεί τα οφέλη και των σχεσιακών και των αντικειμενοστραφών DBMS αποφεύγοντας τα μειονεκτήματά τους. Αυτά τα συστήματα είναι γνωστά υπό τον όρο *object-oriented relational DBMS*. Τα περισσότερα GIS λειτουργούν σε βασικές ή εκτεταμένες σχεσιακές DBMS για την αναπαράσταση των χωρικών αντικειμένων και των attributes οπότε δεν θα αναφερθούμε περαιτέρω στις αντικειμενοστραφής DBMS.

Για την αναπαράσταση των χωρικών αντικειμένων από σχεσιακές DBMS (βλ. Εικόνα 10), έχουν προσδιοριστεί οι ακόλουθες προσεγγίσεις:



Εικόνα 10: Βάσεις δεδομένων σε GIS

2.4.3. Αναπαράσταση από σχεσιακές DBMS

Η πιο στοιχειώδης προσέγγιση είναι να διαμορφωθούν τα χωρικά αντικείμενα από μία συμβατική σχεσιακή DBMS όπως φαίνεται στην Εικόνα 10a. Το μειονέκτημα αυτής της προσέγγισης είναι ότι κανένας αποκλειστικός (dedicated) τύπος δεδομένων για την αντιπροσώπευση των χωρικών αντικειμένων δεν είναι διαθέσιμος, αλλά μόνο απλοί τύποι όπως *integer* και *string*. Κατά συνέπεια, τα χωρικά αντικείμενα πρέπει να συντεθούν από τις σχέσεις των αρχείων μεταξύ των πολυάριθμων πινάκων. Ένα άλλο πρόβλημα προκύπτει από το γεγονός ότι ο αριθμός των σημείων στα χωρικά αντικείμενα συνήθως δεν καθορίζεται, και ως εκ τούτου είναι απαραίτητο να αντιπροσωπευθεί κάθε ένα από αυτά από διάφορα αρχεία αμέσως. Το όφελος αυτής της προσέγγισης είναι ότι οι συμβατικές γλώσσες DBMS και οι query γλώσσες όπως η SQL μπορούν να χρησιμοποιηθούν, αλλά πάσχει από δυσκολίες στη διαχείριση και την επεξεργασία των δεδομένων καθώς επίσης και από την κακή απόδοση λόγω της σύνδεσης πολλών εγγραφών.

2.4.4. Loosely coupled προσέγγιση

Η loosely coupled προσέγγιση αποτελείται από δύο υποσυστήματα, ένα για την αποθήκευση των attributes και ένα άλλο για τη διατήρηση των χωρικών αντικειμένων (Εικόνα 10b). Το πρώτο δίνεται από μια σχεσιακή DBMS, ενώ για το δεύτερο χρησιμοποιείται ένα σύστημα αρχείων. Τα χωρικά αντικείμενα αποθηκεύονται με ένα ορισμένο σχήμα στα αρχεία, και συνδέονται με τα αντίστοιχα περιγραφικά attributes στη σχεσιακή DBMS μέσω εσωτερικών προσδιοριστικών. Αν και αυτή η προσέγγιση είναι καταλληλότερη για την αποθήκευση και την επεξεργασία χωρικών αντικειμένων από την αποκλειστική χρήση μιας σχεσιακής DBMS, πάσχει όμως από δυσκολίες σχετικά με τις τεχνικές αποκατάστασης, την αναζήτηση, και τη βελτιστοποίηση.

2.4.5. Integrated προσέγγιση

Οι εκτεταμένες σχεσιακές DBMS παρέχουν μια ολοκληρωμένη προσέγγιση, με την εισαγωγή νέων χωρικών τύπων δεδομένων όπως σημεία, γραμμές (polylines) και πολύγωνα, καθώς επίσης και τροποποιημένες γλώσσες ερωταπαντήσεων (querying). Με τον τρόπο αυτό διευκολύνεται η υποβολή ερωτημάτων με κριτήριο τα attributes των χωρικών αντικειμένων και η μετέπειτα αποθήκευση των αντίστοιχων αποτελεσμάτων. Έτσι τα χωρικά στοιχεία αντιμετωπίζονται πολύ αποτελεσματικά και μπορεί να βελτιστοποιηθεί η διαδικασία αναζήτησης και επιλογής τους. Αυτή η ολοκληρωμένη προσέγγιση παρουσιάζεται στην Εικόνα 10c.

ΚΕΦΑΛΑΙΟ 3

ΠΛΟΗΓΗΣΗ ΣΕ ΕΣΩΤΕΡΙΚΟΥΣ ΧΩΡΟΥΣ

3.1. Γενικά

Η παρούσα εργασία σχετίζεται με την υλοποίηση μιας εφαρμογής πλοήγησης σε ένα εσωτερικό χώρο (indoor navigation), οπότε σε αυτήν την ενότητα θα αναφερθούμε στις τεχνολογίες με τις οποίες μπορεί να εντοπισθεί η θέση κάποιας κινητής συσκευής (χρήστη) μέσα σε ένα τέτοιο χώρο. Είναι σαφές ότι η πλοήγηση δεν είναι εφικτή εάν δεν αναπτυχθεί και υλοποιηθεί ένα αξιόπιστο σύστημα καθορισμού θέσης (positioning system). Για τον υπολογισμό της θέσης χρησιμοποιείται τριγωνομετρία. Οι πιο γνωστές τεχνολογίες εντοπισμού του χρήστη αναλύονται παρακάτω:

3.2. Τεχνολογία Infrared (IR)

Η ανωτέρω τεχνολογία είναι φθηνή και διαθέσιμη σε ευρύ φάσμα κινητών συσκευών από PDAs έως κινητά τηλέφωνα και φορητούς υπολογιστές. Για το λόγο αυτό έχουν σχεδιαστεί αρκετά συστήματα καθοδήγησης βασισμένα στη συγκεκριμένη τεχνολογία. Για τη σωστή λειτουργία των συστημάτων αυτών απαιτείται οπτική επαφή μεταξύ πομπού και δέκτη (απευθείας επικοινωνία) και απουσία έντονου φωτισμού. Η τεχνολογία αυτή είναι σχετικά ακριβής δεδομένου ότι η εμβέλεια του πομπού, η οποία εξαρτάται από την ισχύ της μπαταρίας και τον αριθμό των ενσωματωμένων LED, είναι μερικά μόνο μέτρα.

Δυστυχώς όμως η αρχική εγκατάσταση (setup) και η συντήρηση ενός συστήματος καθορισμού θέσης ευρείας κλίμακας βασισμένου σε αυτή την τεχνολογία είναι εξαιρετικά δύσκολη καθώς θα πρέπει να εγκατασταθεί μεγάλο πλήθος πομπών των οποίων οι μπαταρίες θα πρέπει να αλλάζονται ανά τακτά σχετικά χρονικά διαστήματα. Αντιθέτως όμως είναι δυνατόν να χρησιμοποιηθεί η IR τεχνολογία σε συγκεκριμένες θέσεις, όπως σημεία ενδιαφέροντος (points of interest) και περιορισμένης έκτασης περιοχές όπου οι υπόλοιπες τεχνολογίες δεν είναι επαρκείς.

3.3. Τεχνολογίες Radio Frequency (RF)

Οι ανωτέρω τεχνολογίες είναι επίσης ευρέως διαδεδομένες κυρίως γιατί μπορούν να ανταπεξέλθουν στα μειονεκτήματα χρήση της IR. Τα RF σήματα έχουν τη δυνατότητα να διαπερνούν τοίχους και εμπόδια, με αποτέλεσμα να παρέχουν καλύτερη κάλυψη και επομένως να απαιτούνται λιγότεροι RF δέκτες για την εγκατάσταση ενός αντίστοιχου συστήματος για την κάλυψη ενός χώρου (π.χ. κτήριο).

Στα μειονεκτήματα χρήσης της τεχνολογίας αυτής είναι η υψηλότερη απαίτηση σε κατανάλωση ισχύος και η δυσκολία απόλυτα κατευθυντικής επικοινωνίας (directed communication). Επιπρόσθετα η χρήση RF συχνοτήτων απαιτεί αδειοδότηση. Τα συστήματα υψηλής ακριβείας (περίπου 10-30 cm) χρησιμοποιούν Time Difference of Arrival (TDOA) ή Angle of Arrival (AOA), αλλά για τη λειτουργία τους απαιτείται η διατήρηση συγχρονισμού με τους δέκτες (σταθμούς βάσης). Το μειονέκτημα αυτών των συστημάτων είναι το υψηλό κόστος τους. Αντιθέτως τα συστήματα που στηρίζουν τη λειτουργία τους στη μέτρηση της ισχύος του σήματος είναι λιγότερο πολύπλοκα και άρα φθηνότερα αλλά υστερούν σε ακρίβεια (περίπου 1-3 m).

Μια ενδιάμεση λύση είναι η χρήση ενός συνδυασμού RF τεχνολογιών ανάλογα με την ακρίβεια ανά χώρο που επιθυμούμε. Ενδεικτικά αναφέρεται η λύση χρήσης συγχρόνως τεχνολογιών Wireless LAN (λιγότερο ακριβής αλλά φθηνή) και Ultra Wide Band (UWB) (μεγαλύτερη ακρίβεια και κόστος), για τον καθορισμό θέσης. Ο τρόπος χρήσης των τεχνολογιών αυτών περιγράφεται παρακάτω.

3.3.1. Wireless LAN

Η τεχνολογία WLAN χρησιμοποιεί, για τον υπολογισμό της θέσης, την ισχύ του σήματος των access points σε μια περιοχή. Υπάρχουν δύο διαφορετικές προσεγγίσεις για τον καθορισμό της θέσης. Σύμφωνα με το εμπειρικό μοντέλο, οι μετρούμενες τιμές ισχύος σήματος συγκρίνονται με τις τιμές ισχύος που έχουν προηγουμένως καταχωρηθεί σε μια βάση δεδομένων. Επομένως είναι σαφές ότι θα πρέπει να έχει προηγηθεί μια φάση βαθμονόμησης προκειμένου να δημιουργηθεί ένας χάρτης με τιμές ισχύος ανά θέση (radio map) ο οποίος αποτελεί τη βάση δεδομένων. Η δεύτερη προσέγγιση ονομάζεται μοντέλο διάδοσης και βασίζεται σε μαθηματικές εξισώσεις οι οποίες περιγράφουν την απώλεια της ισχύος του σήματος στο χώρο. Χρησιμοποιώντας τις εξισώσεις αυτές είναι δυνατόν να υπολογιστεί μια κατά προσέγγιση απόσταση της WLAN κάρτας από το access point και μέσω τριγωνομετρίας να βρεθεί η απόλυτη θέση.

Η χρήση τυχόν υφιστάμενης WLAN υποδομής καθιστά τη λύση αυτή φθηνή και ευέλικτη. Για τον εντοπισμό θα πρέπει να χρησιμοποιηθεί μια συσκευή η οποία θα εκτελεί μια client εφαρμογή η οποία θα διαβάζει και θα στέλνει την ισχύ του σήματος σε μια «Μηχανή Εύρεσης Θέσης» που θα είναι υπεύθυνη για τη συλλογή των στοιχείων από τις κινητές συσκευές και την εύρεση της θέσης τους.

3.3.2. Ultra Wide Band (UWB)

Ένα σημαντικό πρόβλημα στη χρήση RF για εύρεση θέσης είναι η παραμόρφωση πολλαπλών διαδρομών (multipath distortion) των RF σημάτων, η οποία οδηγεί σε μειωμένη ακρίβεια στην εύρεση θέσης. Το πρόβλημα αμβλύνεται με χρήση UWM παλμούς οι οποίοι καθιστούν εφικτό, λόγω της μικρής διάρκειας-εύρους (duration) των παλμών, το φιλτράρισμα των ανακλώμενων σημάτων από τα αρχικά και έτσι στην επίτευξη ακρίβειας εύρεσης της τάξης των 15cm.

3.4. Άλλες Τεχνολογίες

Επιπρόσθετα υπάρχει ένας αριθμός οπτικών συστημάτων τα οποία βασίζονται στην επεξεργασία εικόνας. Η ιδέα των συστημάτων αυτών είναι η μίμηση των ικανοτήτων του ανθρώπινου εγκεφάλου και των ματιών. Μέσω τεχνικών εύρεσης ακμών (edge detection) ή αναγνώρισης αντικειμένων από μια ανεπεξέργαστη (raw) εικόνα και σύγκρισης τους με αντίστοιχα αντικείμενα μιας βάσης δεδομένων εικόνων, είναι δυνατός ο υπολογισμός της απόστασης από καλώς ορισμένα αντικείμενα και επομένως της απόλυτης θέσης της κάμερας. Η επεξεργασία εικόνας όμως είναι μια εργοβόρα υπολογιστικά διαδικασία και απαιτεί χρήση κάμερας υψηλής ακρίβειας καθιστώντας τα αντίστοιχα συστήματα εξαιρετικά ακριβά (με χρήση υψηλής ακρίβειας καμερών και αποκλειστικής χρήσης server για επεξεργασία εικόνας) ή όχι αρκετά ακριβή στον υπολογισμό της θέσης (κάνοντας χρήση φθηνών καμερών και φορητών συσκευών μικρής επεξεργαστικής ισχύος για επεξεργασία εικόνας).

Στην παρούσα φάση αυξανόμενο ενδιαφέρον παρουσιάζουν τα συστήματα indoor GPS αλλά παρουσιάζουν τεχνικές δυσκολίες στην υλοποίηση. Επιπλέον συστήματα εύρεσης θέσης βασιζόμενα σε RFID τεχνολογία έχουν αρχίσει να αντικαθιστούν τα αντίστοιχα IR.

3.5. Data Format Types

Υπάρχουν τρεις τύποι απεικόνισης δεδομένων (data mapping) και μορφοποίησης GIS δεδομένων. Οι τύποι αυτοί μαζί με κάποια παραδείγματα υλοποίησης τους φαίνονται παρακάτω:

- File-based – Shapefiles, GeoTIFF images, Microstation Design Files (DGN)
- Directory-based: ESRI ArcInfo Coverages, US Census Tiger
- Database connections: PostGIS, ESRI ArcSDE, MySQL

Κάθε τύπος δεδομένων αποτελείται από την πηγή δεδομένων (data source) και από ένα ή περισσότερα επίπεδα (layers). Για την πηγή δεδομένων και το layer δίδονται οι εξής ορισμοί:

Data Source: Μία ομάδα layers που είναι αποθηκευμένη σε ένα κοινό χώρο. Ο χώρος αυτός μπορεί να είναι ένα αρχείο το οποίο περιλαμβάνει τα layers ή ένα φάκελος (folder) που έχει ένα σύνολο αρχεία που χαρακτηρίζουν τα layers.

Layer: Ένα υποσύνολο του data source που συχνά περιέχει πληροφορία ενός τύπου ανυσματικής μορφής (σημείο, γραμμή, πολύγωνο).

3.5.1. File-based Δεδομένα

Τα File-based δεδομένα αποτελούνται από ένα ή περισσότερα αρχεία που είναι αποθηκευμένα σε κάποιο φάκελο (folder). Σε πολλές περιπτώσεις χρησιμοποιείται ένα μόνο αρχείο (π.χ. DGN), αλλά δεν αποκλείεται να χρησιμοποιούνται και περισσότερα. Για παράδειγμα στην περίπτωση των ESRI Shapefiles χρησιμοποιούνται τρία αρχεία, καθένα από τα οποία έχει και διαφορετικό extension (SHP, DBF, SHX). Η ύπαρξη και των τριών αρχείων είναι απαραίτητη καθώς το καθένα επιτελεί διαφορετικό ρόλο.

Τα ονόματα αρχείων συνήθως αποτελούν και το όνομα των πηγών δεδομένων. Τα αρχεία αυτά περιλαμβάνουν layers τα οποία ενδέχεται να είναι προφανή από το όνομα του αρχείου χωρίς αυτό να είναι απαραίτητο. Για παράδειγμα στα Shapefiles, υπάρχει ένα data source για κάθε shapefile και ένα layer που έχει το ίδιο όνομα με αυτό του αρχείου shapefile.

3.5.2. Directory-based Δεδομένα

Τα directory-based δεδομένα αποτελούνται από ένα ή περισσότερα αρχεία τα οποία είναι αποθηκευμένα με συγκεκριμένο τρόπο μέσα σε ένα φάκελο γονέα. Σε κάποιες περιπτώσεις (π.χ. Coverages) μπορεί να είναι επίσης απαραίτητη η ύπαρξη επιπλέον φακέλων σε άλλα σημεία του δένδρου αρχείων (file tree) προκειμένου να είναι δυνατή η πρόσβαση στα directory-based δεδομένα. Το ίδιο το directory ενδέχεται να είναι το data source. Διαφορετικά αρχεία μέσα στο directory συχνά αντιπροσωπεύουν τα διαθέσιμα layers δεδομένων.

Για παράδειγμα τα ESRI ArcInfo Coverages αποτελούνται από περισσότερα του ενός αρχεία με επέκταση αρχείου ADF μέσα σε ένα φάκελο. Το αρχείο «PAL.ADF» αντιπροσωπεύει δεδομένα τύπου πολύγωνο. Το «ARC.ADF» περιλαμβάνει δεδομένα τύπου τόξου (arc) ή γραμμής. Ο φάκελος αποτελεί το data source και κάθε αρχείο ADF είναι ένα layer.

3.5.3. Database Connections

Οι database connections είναι παρόμοιες με τις δομές file-based και directory-based, που αναφέρονται παραπάνω, υπό την έννοια ότι παρέχουν δεδομένα γεωγραφικών συντεταγμένων στον Mapserver ώστε να χρησιμοποιηθούν για τη δημιουργία των ανυσματικών συνόλων δεδομένων.

Οι database connections παρέχουν μια ροή δεδομένων γεωγραφικών συντεταγμένων η οποία αποθηκεύεται προσωρινά (π.χ. στη μνήμη) και στη συνέχεια διαβάζεται από τον Mapserver προκειμένου να δημιουργηθεί ο χάρτης. Είναι δυνατόν να απαιτούνται και επιπρόσθετα δεδομένα (π.χ. attribute) αλλά ο πυρήνας είναι τα δεδομένα γεωγραφικών συντεταγμένων.

Θα πρέπει εδώ να επισημανθεί μια σημαντική διάκριση μεταξύ των βάσεων δεδομένων. Οι βάσεις δεδομένων στις οποίες αναφερόμαστε εδώ είναι οι χωρικές (spatial) οι οποίες περιλαμβάνουν γεωγραφικά δεδομένα σε μια συγκεκριμένη δική τους δομή. Αυτό διαφέρει από τις βάσεις δεδομένων που είναι αυστηρά με τη μορφή πινάκων (tabular databases) οι οποίες δεν μπορούν να περιλαμβάνουν γεωγραφικές συντεταγμένες με τον ίδιο τρόπο. Κάποια πολύ απλά γεωγραφικά δεδομένα είναι δυνατόν να περιληφθούν απλές βάσεις δεδομένων αλλά για οτιδήποτε πιο σύνθετο απαιτείται η χρήση χωρικών βάσεων δεδομένων. Υπάρχουν πολλές επεκτάσεις (extensions) βάσεων δεδομένων για απεικόνιση χωρικών δεδομένων, τόσο εμπορικού τύπου όσο και ανοικτού κώδικα. Από τις πλέον ισχυρές επεκτάσεις είναι η PostGIS η οποία παρέχει δυνατότητα αποθήκευσης χωρικών δεδομένων στη βάση δεδομένων PostgreSQL. Με τη χρήση της PostGIS μπορούμε όχι μόνο να αποθηκεύσουμε χωρική πληροφορία αλλά και να χειριστούμε τα χωρικά δεδομένα με SQL εντολές. Μια άλλη βάση δεδομένων με χωρικές δυνατότητες είναι η MySQL.

Οι database connections συνήθως αποτελούνται από τα παρακάτω μέρη:

Host: Directories στον server ή στον υπολογιστή που φιλοξενεί τη βάση δεδομένων.

Όνομα βάσης δεδομένων: Το όνομα της βάσης δεδομένων στο οποίο επιθυμούμε να έχουμε πρόσβαση και η οποία τρέχει στον host.

Όνομα χρήστη/κωδικοί: Τα δικαιώματα πρόσβασης συνήθως προσδιορίζονται ανά χρήστη. Η πρόσβαση σε συγκεκριμένα τμήματα των γεωγραφικών δεδομένων συνήθως απαιτεί:

Όνομα Table/View: Το όνομα του πίνακα ή της όψης που περιέχει τα ζητούμενα δεδομένα

Όνομα στήλης γεωγραφικών δεδομένων: Εκεί όπου η γεωμετρία ή οι συντεταγμένες είναι αποθηκευμένες

ΚΕΦΑΛΑΙΟ 4

ΣΥΓΚΡΙΣΗ VECTOR-RASTER ΓΙΑ ΕΦΑΡΜΟΓΕΣ ΔΙΑΔΙΚΤΥΟΥ

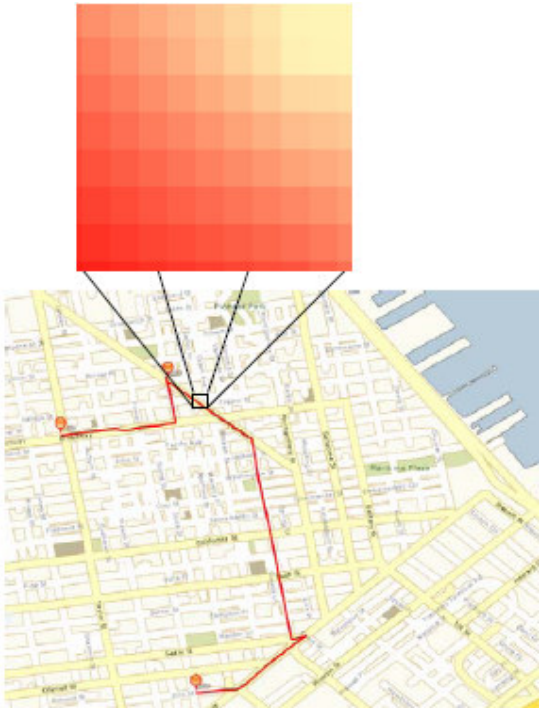
4.1. Raster Εικόνες

Η raster εικόνα αποτελείται από pixels. Το pixel είναι το μικρότερο στοιχείο της εικόνας. Για παράδειγμα μια ψηφιακή φωτογραφία αποτελεί μια raster εικόνα. Οι raster εικόνες έχουν συγκεκριμένο αριθμό pixels ο οποίος περιγράφεται από τον όρο ανάλυση. Επειδή κάθε raster εικόνα έχει συγκεκριμένη ανάλυση υπάρχουν όρια στον τρόπο χρήσης τους. Για παράδειγμα έστω μια εικόνα με ανάλυση 2000x1600=3,2 Mpixels την οποία θέλουμε να εκτυπώσουμε με μια καλή ανάλυση της τάξης των 200ppi (pixels per inch). Αυτό σημαίνει ότι οι διαστάσεις της εικόνας που θα εκτυπωθεί θα είναι:

H=2000/200=10 inches.

W=1600/200=8 inches.

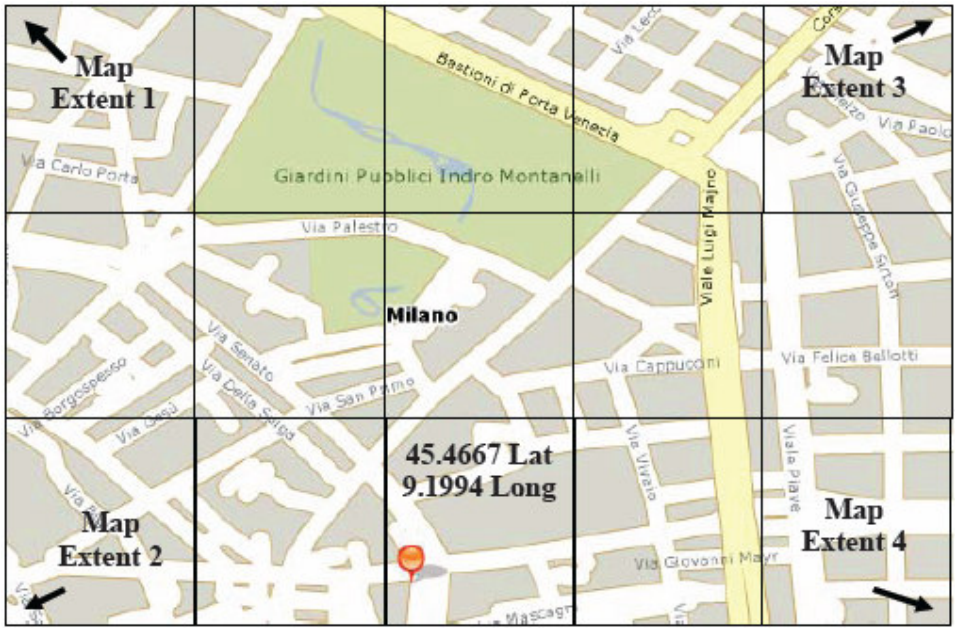
Όπως φαίνεται στην Εικόνα 11, οι raster χάρτες αποτελούνται από ένα δυσδιάστατο πίνακα (γραμμές και στήλες) χρωματισμένων pixels τα οποία σχηματίζουν την εικόνα του χάρτη. Σε υψηλά επίπεδα μεγέθυνσης οι raster εικόνες εμφανίζουν ακανόνιστες ακμές οι οποίες είναι οι άκρες των μεμονωμένων pixel. Αυτή η κοκκοποίηση συχνά αναφέρεται ως “pixelation” και εμφανίζεται όταν τα μεμονωμένα στοιχεία από τα οποία αποτελείται η εικόνα γίνονται ορατά στο μάτι. Οι εικόνες αυτές εμφανίζονται κανονικά όταν παρατηρούνται στην αιτηθείσα κλίμακα αλλά σε περίπτωση που η εικόνα μεγεθυνθεί τα μεμονωμένα pixels γίνονται εμφανή.



Εικόνα 11: Raster γραφικά που αποτελούνται από pixels με χρώμα

Το 2005 ξεκίνησε η παροχή χαρτών στη μορφή των tiled raster εικόνων. Ολόκληρη η εικόνα αποτελείτο από πολλές μικρότερες εικόνες οι οποίες δημιουργούσαν τον χάρτη που έβλεπε κανείς στον browser. Όλες οι εικόνες είχαν δημιουργηθεί ή προ-παραχθεί σε κάποια στιγμή πρωθύστερη του αιτήματος, από την mapping service, για προβολή τους. Το αποτέλεσμα ήταν να επιτυγχάνεται υψηλή απόδοση, καθώς ο server δε χρειαζόταν χρόνο για τη δημιουργία του χάρτη αλλά απλώς ανακαλούσε τα κατάλληλα αρχεία raster και τα έστελνε στον client. Επιπρόσθετα αυτή η τεχνική επέτρεπε ομαλή πλοήγηση (panning) στο χάρτη. Από την άλλη όμως, επειδή οι εικόνες ήταν συγκεκριμένες υπήρχε περιορισμός στον τρόπο χειρισμού της απεικόνισης του χάρτη. Για παράδειγμα τα map styles που καθορίζουν τα χρωματικά μοτίβα που θα χρησιμοποιηθούν για το χάρτη δεν μπορούσαν να μεταβληθούν. Τυπικά τα μόνα μετα-δεδομένα που είναι διαθέσιμα στον δημιουργό χαρτών raster είναι τα όρια (extents) γεωγραφικού μήκους και πλάτους όπως φαίνεται στην Εικόνα 12. Τα όρια αυτά είναι το ελάχιστο ορθογώνιο παραλληλόγραμμο που περιέχει το χάρτη και ορίζεται από 4 ζεύγη (ένα για κάθε γωνία του παραλληλογράμμου) γεωγραφικών συντεταγμένων μήκους και πλάτους. Αν ο σχεδιαστής γνωρίζει τα όρια αυτά μπορεί να χρησιμοποιήσει την εικόνα του χάρτη ως στατικό background καμβά για να τοποθετήσει αντικείμενα όπως σημεία ένδειξης πάνω από την εικόνα-χάρτη. Τα σημεία ένδειξης ευθυγραμμίζονται στο χάρτη

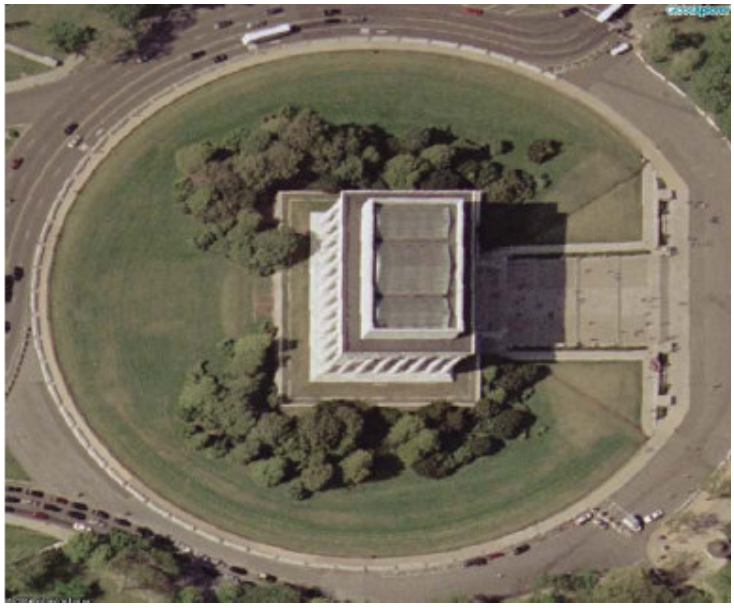
μέσω κάποιας συνάρτησης της εφαρμογής η οποία μετατρέπει τις γεωγραφικές συντεταγμένες μήκους και πλάτους σε pixels της εικόνας του χάρτη.



Εικόνα 12: Προσομοίωση tiled Raster Χάρτη με τα όρια των Tiles και του Χάρτη να διακρίνονται

Η χρήση raster εικόνων σε χάρτες γίνεται για διάφορους λόγους. Καθένας μπορεί να έχει πρόσβαση μέσω internet σε raster γραφικά χωρίς να υπάρχει απαίτηση για ειδικά plug-ins. Ο τελικός χρήστης μπορεί να έχει πρόσβαση σε εφαρμογές που περιέχουν raster εικόνες χωρίς να απαιτείται να εγκαταστήσει στο σύστημά του κάποιο ειδικό λογισμικό. Υπάρχουν χρήστες που για λόγους ασφαλείας αποφεύγουν την εγκατάσταση plug-ins και έτσι μόνο εφαρμογές που χρησιμοποιούν raster εικόνες είναι προσβάσιμες από όλους τους χρήστες του διαδικτύου. Επιπρόσθετα η δημιουργία των raster εικόνων γίνεται αποκλειστικά στον mapping server με αποτέλεσμα να μειώνεται ο κίνδυνος της μη απεικόνισης του χάρτη λόγω της επεξεργαστικής δυνατότητας του client.

Καθένα από τα είδη raster (GIF, JPEG και PNG) έχει τα πλεονεκτήματά του. Το GIF είναι ο βέλτιστος τύπος για απεικόνιση δρόμων ή χαρτών με γραμμές και επιπρόσθετα υποστηρίζει διαφάνεια (transparency). Το JPEG δημιουργεί το μικρότερο σε όγκο αρχείο όταν χρησιμοποιείται σε χάρτες με εικόνες και υποστηρίζει πάνω από 16 εκατομμύρια αποχρώσεις (Η **Εικόνα 13** παρέχει ένα παράδειγμα ενός χάρτη με εικόνες). Το PNG που έχει δημιουργηθεί ως ένας τύπος, χωρίς απαιτήσεις για πνευματικά δικαιώματα, ισοδύναμος με τον GIF, όταν ο GIF ήταν πατενταρισμένος, υποστηρίζει διαφάνεια και είναι μη απωλεστικός, που σημαίνει ότι δεν υπάρχει απώλεια πληροφορίας κατά τη συμπίεση.



Εικόνα 13: Ο JPEG είναι καλή επιλογή για απεικόνιση εικόνων από δορυφόρο

4.2. Ανυσματικές Εικόνες

Αντιθέτως με τις raster εικόνες οι ανυσματικές (vector graphics) είναι ανεξάρτητες από την ανάλυση. Ένα ανυσματικό γραφικό ορίζεται με μαθηματικούς όρους που σημαίνει ότι είναι απείρως αυξομειώσιμο. Στις εφαρμογές της Adobe οι μαθηματικές γραμμές ορίζονται ως καμπύλες Bezier. Τα ευθύγραμμα τμήματα σε μια Bezier ενώνονται με anchor σημεία. Κάθε σημείο anchor μπορεί να έχει μια επιπρόσθετη εφαπτόμενη γραμμή που ονομάζεται χειριστής διεύθυνσης και ορίζει πώς το ευθύγραμμο τμήμα καμπυλώνει.

Τα vector γραφικά δημιουργούνται με αυτόματο τρόπο αντίθετα με τα raster που δημιουργούνται μέσω σάρωσης ή από μια ψηφιακή μηχανή. Υπάρχουν εργαλεία μετατροπής raster σε vector αλλά συνήθως τα vector δημιουργούνται εξ αρχής.

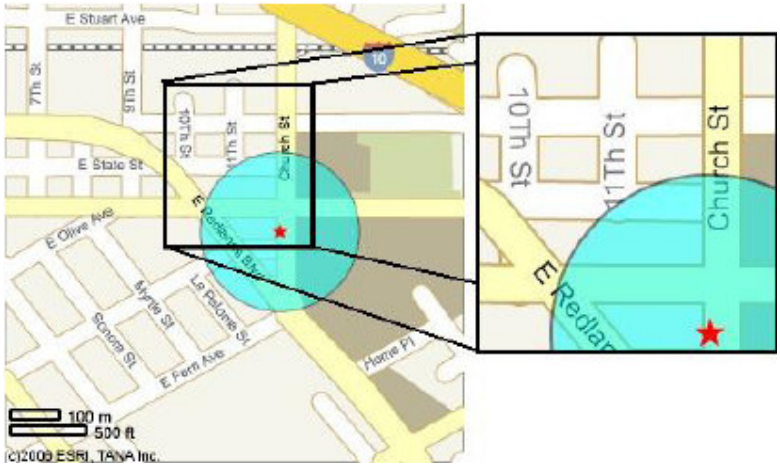
Η χρήση ανυσματικών γραφικών οδηγεί τις εφαρμογές πέραν της απλής απεικόνισης χαρτών παρέχοντας μια πλούσια διαδραστική εμπειρία στον client ο οποίος χρησιμοποιεί χαρακτηριστικά (features) μέσα στα δεδομένα του χάρτη. Οι πιο γνωστοί και ευρέως χρησιμοποιούμενοι τύποι ανυσματικών γραφικών είναι ο Adobe Flash (SWF) και ο πατενταρισμένος Scalable Vector Graphics (SVG). Λόγω του ότι τα ανυσματικά γραφικά αποτελούνται από οδηγίες που προσδιορίζουν μαθηματικά σχήματα, οι σχεδιαστές μπορούν να δημιουργήσουν εφαρμογές που χειρίζονται ολόκληρα αντικείμενα του χάρτη. Τα ανυσματικά γραφικά εμφανίζονται με την ίδια μορφή ανεξάρτητα εάν ζητηθεί μεγέθυνση ή σμίκρυνσή τους και επιτρέπουν το

δυναμικό επανασχεδιασμό τους χωρίς να απαιτείται κλήση για νέα εικόνα από τον map server.

Για παράδειγμα για τον ορισμό μιας ακτίνας γύρω από συγκεκριμένη γεωγραφική συντεταγμένη στο χάρτη, η εφαρμογή θα πρέπει να εισάγει μια γραμμή κώδικα όμοια με την παρακάτω μέσα σε ένα αρχείο svg. Όταν ο client εμφανίσει την εικόνα ο κύκλος θα εμφανιστεί στο χάρτη:

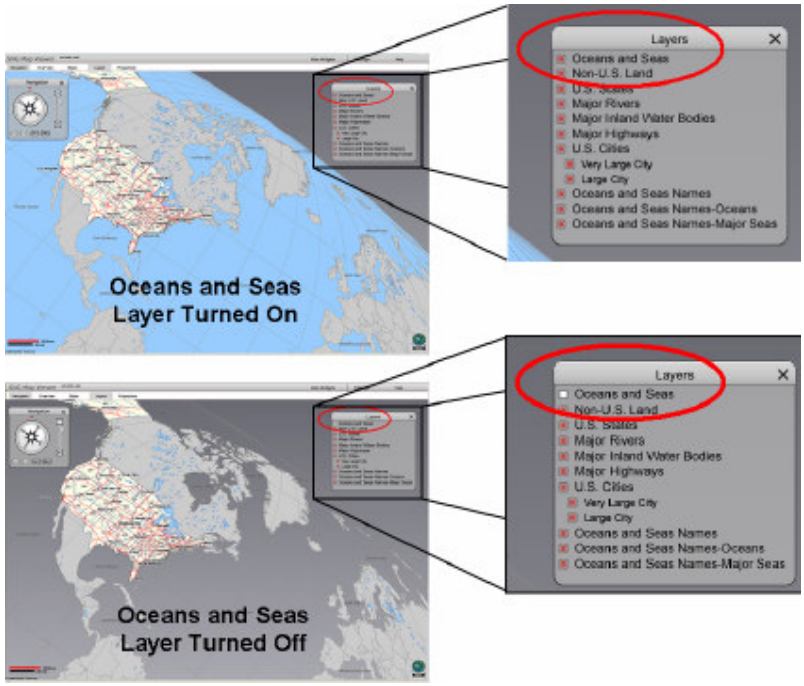
<circle cx=’103’ cy=’103’ r=’50’ style=’fill:cyan; opacity:0.3; stroke:black’ />

Ο παραπάνω κώδικας προσθέτει ένα ημιδιάφανο κυανό κύκλο ακτίνας 50 με κέντρο τη θέση (103,103) περιγράμματος χρώματος μαύρου, όπως φαίνεται στην Εικόνα 14.



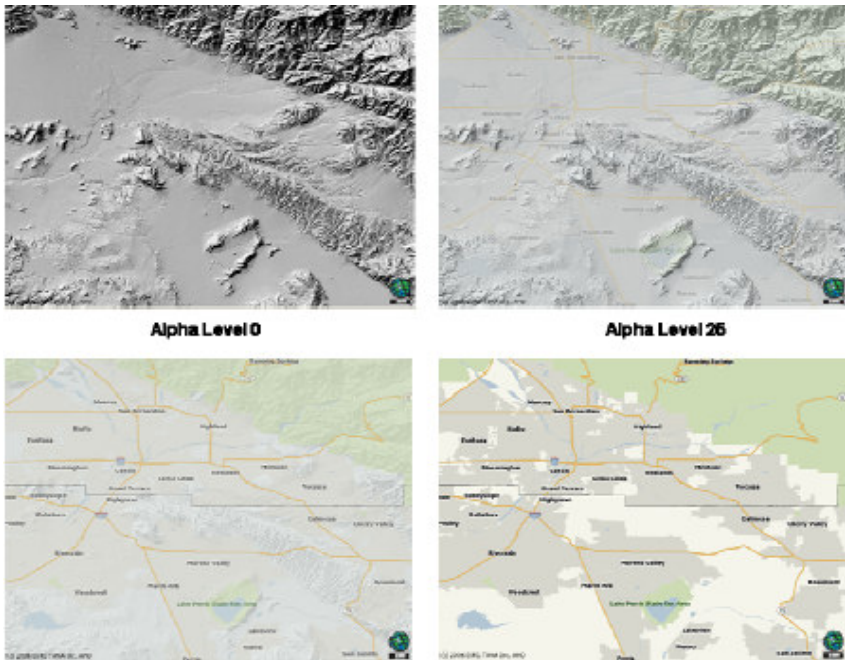
Εικόνα 14: Τα ανυσματικά γραφικά αποτελούνται από εντολές γεωμετρίας

Η Εικόνα 15 απεικονίζει τον SVG Map Viewer των ArcWeb Services όπου φαίνεται μια εφαρμογή που χρησιμοποιεί τη λειτουργικότητα των SVG για να βελτιώσει το αποτέλεσμα που λαμβάνει ο τελικός χρήστης. Ανάμεσα σε άλλες λειτουργίες ο χρήστης μπορεί να θέσει on ή off τα layers, να μετακινήσει τη γη υπό διαφορετικές γωνίες και να αλλάξει την προβολή του χάρτη.



Εικόνα 15: Εικόνες πριν και μετά: Ο χρήστης θέτει layer και σχήματα on και off

Οι εφαρμογές που χρησιμοποιούν ανυσματικά γραφικά είναι γρήγορες για πολλούς λόγους. Οι πιο βασικές λειτουργίες που είναι η εμφάνιση ή απόκρυψη των επιπέδων με γεωγραφικά δεδομένα καθώς και η μεγέθυνση ή σμίκρυνση, μπορούν να γίνουν τοπικά χωρίς την ανάγκη να γίνει νέα κλήση στον server. Όπως φαίνεται στην Εικόνα 16, με το που θα ανακληθούν τα δεδομένα από την εφαρμογή όπως ο ArcWeb Service Explorer, ο τελικός χρήστης μπορεί να αλλάξει τη μορφή του χάρτη ή άλλα χαρακτηριστικά του on the fly. Για αυξημένη λειτουργικότητα όπως η χωρική αναζήτηση, η πληροφορία μπορεί να προστεθεί εύκολα ως ένα layer αναζήτησης στην εφαρμογή του χάρτη το οποίο θα παρέχεται από το API της Web service.



Εικόνα 16: Αλλάζοντας τη διαφάνεια των layers on the fly

Ο Πίνακας 1 δίνει μια επισκόπηση των διαφορών ανάμεσα στους τύπους SWF και SVG. Ο SWF ο οποίος μερικές φορές αναφέρεται και ως Flash είναι ένας ιδιοκτησιακός ανυσματικός τύπος γραφικών που δημιουργήθηκε από την Adobe/Macromedia Flash. Τα αρχεία SWF εμφανίζονται από τον Adobe Flash Player ο οποίος μπορεί να είναι είτε αυτόνομος player είτε plug-in στον browser.

Πίνακας 1: Σύγκριση μεταξύ SWF και SVG

	Flash (SWF) Maps	SVG Maps
Ανυσματικοί	ΝΑΙ	ΝΑΙ
Browser Plug-in	Οι περισσότεροι χρήστες ήδη χρησιμοποιούν το plug-in Flash player	Ελάχιστοι χρήστες έχουν ήδη το plug-in
Αδειοδότηση	Adobe proprietary	Non-proprietary
Client-Side Programming Control	Adobe Flash Player ή ενσωματωμένο στο API	Οι περισσότεροι XML parsers
Δυνατότητα τροποποίησης του map file	ΟΧΙ (δυαδικό αρχείο)	ΝΑΙ (XML αρχείο)

Ο τύπος SVG είναι ανοικτού κώδικα προδιαγραφών μη ιδιοκτησιακού καθεστώτος που δημιουργήθηκε από το World Wide Web Consortium, το οποίο έχει δημιουργήσει και τα HTML, CSS και XML Επειδή το SVG είναι ανοικτού κώδικα και βασίζεται στην XML οι περισσότεροι από τους γνωστούς XML parsers μπορούν να διαβάσουν τον SVG κώδικα (αντίθετα με τα flash αρχεία). Σαν αποτέλεσμα το περιεχόμενο των SVG

αρχείων είναι αναγνώσιμο και μπορεί να διαμορφωθεί από γλώσσες προγραμματισμού που αλληλεπιδρούν με XML. Τα SVG αρχεία μπορούν επίσης να ενσωματωθούν σε Adobe PDF έγγραφα. Το χαρακτηριστικό αυτό είναι ιδιαίτερα σημαντικό στις περιπτώσεις που υπάρχει απαίτηση ποιοτικής εκτύπωσης χαρτών ενσωματωμένων σε αναφορές.

Επιπρόσθετα ένα αρχείο SVG μπορεί να γραφεί και διαμορφωθεί από οποιοδήποτε text editor καθώς είναι ανθρώπινα κατανοητή και δεν είναι δυαδική. Επίσης προσφέρει ένα pixel-precise layout το οποίο ταιριάζει με API μεθόδους και εφαρμογές μετατροπής pixel σε συντεταγμένες. Ακόμη έχει τη δυνατότητα χρήσης CSS όσον αφορά τον έλεγχο του layout, στυλ, χρωμάτων και γραμματοσειρών. Τέλος σημειώνεται όμως ότι δεν είναι όλοι οι SVG renderers συμβατοί με CSS.

4.3. Χρήση Raster και Vector Εικόνων για απεικόνιση χαρτών

Παραδοσιακά οι χάρτες μέσω διαδικτύου βασίζονταν σε raster γραφικά τα οποία δημιουργούνταν από έναν Web Server και παραδίδονταν στον φυλομετρητή του εξυπηρετούμενου (client browser). Στις μέρες μας η τάση είναι η δημιουργία εφαρμογών χαρτογραφίας με χρήση ανυσματικών γραφικών.

Οι raster χάρτες δεν απαιτούν κάποιο ειδικό plug-in προκειμένου να απεικονιστούν στον client. Από την άλλη όμως αυτού του είδους οι χάρτες έχουν μικρή λειτουργικότητα. Από τη στιγμή που δημιουργηθεί η εικόνα παραμένει αμετάβλητη. Για την τροποποίησή της απαιτείται επιπλέον κλήση στον map server. Για την επίτευξη γρήγορης απεικόνισης τέτοιων εικόνων καθώς και για ομαλή πλοήγηση (panning) συνήθως ο server έχει δημιουργήσει εκ των προτέρων τις εικόνες και τις αποδίδει αμέσως μετά την κάθε κλήση. Έτσι από τη στιγμή που οι εικόνες δημιουργηθούν στον server μπορούν να χρησιμοποιηθούν μεν πολλές φορές αλλά από την άλλη δεν είναι δυνατόν να τροποποιηθούν. Αντιθέτως οι ανυσματικοί χάρτες μπορούν να προσαρμοστούν ανά πάσα στιγμή επιτρέποντας τη δημιουργία δυναμικών εφαρμογών διαδικτύου όπου οι χρήστες αλληλεπιδρούν με τα χαρτογραφικά δεδομένα.

Στον **Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.** δίδονται περιληπτικά τα χαρακτηριστικά και οι δυνατότητες των χαρτών μορφής raster, tiled raster και ανύσματος (vector).

Πίνακας 2: Σύγκριση λειτουργικότητας Raster και Vector Χαρτών

Μορφή Χάρτη	Δυνατότητα εναλλαγής	Κατάλληλο για click,	Προσδιορισμός των Features	Highlight, Animate	Απαίτηση για χρήση

	layer, style, projection	Drag and Pan		Features	ειδικού plug-in
Raster (JPEG, GIF, PNG)	NAI	OXI	Απαιτεί νέα κλήση (request) σε server	OXI	OXI
Tiled Raster (overlaid JPEG, GIF, PNG)	OXI	NAI	OXI	OXI	OXI
Vector (SWF, SVG)	NAI	NAI	Δεν απαιτεί νέα κλήση (request) σε server	NAI	NAI

Παραδοσιακά οι χάρτες στο διαδίκτυο δημιουργούνταν στον server και παραδίδονταν στον client με τη μορφή γραφικών τύπου μορφής JPEG ή GIF. Εάν ο χρήστης επιθυμούσε να χειριστεί το χάρτη (για παράδειγμα ζητούσε επιπλέον layer) η client εφαρμογή έκανε νέα κλήση στον server ο οποίος δημιουργούσε ένα νέο χάρτη.

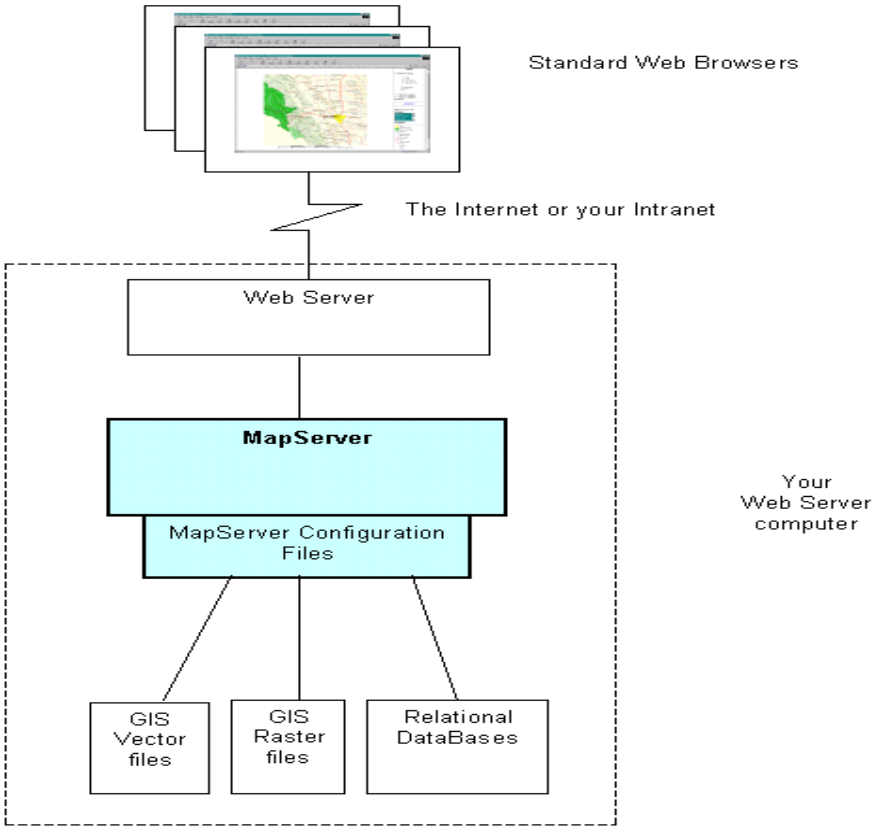
ΚΕΦΑΛΑΙΟ 5

MAPSERVER

5.1. Mapserver overview

Ο server με τον οποίο θα υλοποιηθεί η εφαρμογή μας είναι ο Mapserver [7] [8]. Ο MapServer είναι ένα περιβάλλον ανάπτυξης open source για τη δημιουργία χωρικών εφαρμογών στο internet. Ο Mapserver δεν είναι ένα ολοκληρωμένο GIS σύστημα και ούτε αυτή είναι η επιδίωξή του. Όμως ο Mapserver υπερέχει στην απόδοση των χωρικών δεδομένων (χάρτες, εικόνες, και διανυσματικά στοιχεία) για το Διαδίκτυο. Εκτός από το browsing των GIS δεδομένων, ο MapServer επιτρέπει τη δημιουργία "γεωγραφικών χαρτών", δηλαδή χαρτών που μπορούν να κατευθύνουν τους χρήστες σε κάποιο σημείο ενδιαφέροντος.

Ο MapServer αναπτύχθηκε αρχικά από το πανεπιστήμιο της Μινεσότα (UMN) [9] στα πλαίσια του προγράμματος Fornet σε συνεργασία με τη NASA και το Τμήμα Φυσικών Πόρων της Μινεσότα (MNDNR). Τώρα, το MapServer project φιλοξενείται από το TerraSIP πρόγραμμα, το οποίο υποστηρίζεται από τη NASA μεταξύ του UMN και μιας κοινοπραξίας διαχείρισης γης. Η αρχιτεκτονική του Mapserver φαίνεται στην Εικόνα 17.



Εικόνα 17: Αρχιτεκτονική του Mapserver

5.2. Λόγοι επιλογής του Mapserver

Η χρήση του Mapserver έχει αρκετά πλεονεκτήματα. Ένα από αυτά είναι η δυνατότητα που προσφέρει για ευρεία πρόσβαση στην χαρτογραφημένη πληροφορία από πολλούς χρήστες, ιδιαίτερα μέσω του Διαδικτύου. Πολλοί GIS προγραμματιστές πρέπει να δημιουργήσουν χαρτογραφικά προϊόντα για εκείνους που υποστηρίζουν ή την εταιρεία τους. Ο MapServer, δίνει τη δυνατότητα σε απλούς χρήστες να δημιουργήσουν χάρτες χωρίς να χρειάζονται κάποια ιδιαίτερα εργαλεία ή τη βοήθεια των προγραμματιστών. Αυτό μειώνει στη συνέχεια την πίεση στο ειδικευμένο προσωπικό. Άλλοι χρησιμοποιούν τον Mapserver επειδή είναι μια από λίγες διαθέσιμες λύσεις για εκείνους που διαχειρίζονται διαφορετικά data formats. Ο MapServer, μέσω της χρήσης των βιβλιοθηκών όπως το GDAL/OGR, μπορεί να έχει πρόσβαση σε διάφορα data formats χωρίς μετατροπή δεδομένων. Ένας άλλος σημαντικός λόγος χρήσης του είναι το γεγονός ότι είναι ανοικτό λογισμικό (open source).

Θεωρήστε ότι θα μπορούσατε να έχετε μια συλλογή 10 διαφορετικών συνόλων δεδομένων χαρτογράφησης, τα οποία πρέπει να εμφανιστούν στον ίδιο χάρτη ταυτόχρονα χωρίς οποιαδήποτε από αυτά τα δεδομένα να μετατραπούν από το αρχικό

format τους. Τα αρχικά format περιλαμβάνουν εκείνα που χρησιμοποιούνται από τους διαφορετικούς εμπορικούς προμηθευτές. Τα ESRI shapefiles, τα αρχεία σχεδίου Intergraph Microstation (DGN), τα MapInfo TAB αρχεία, και οι Oracle χωρικές βάσεις δεδομένων μπορούν όλες να χαρτογραφηθούν μαζί χωρίς μετατροπή. Άλλα μη εξειδικευμένα formats μπορούν να χρησιμοποιηθούν επίσης, συμπεριλαμβανομένων των προτύπων OGC για τη γλώσσα GML, τον Web Map Server (WMS), τον Web Feature Server (WFS), την PostGIS και άλλες βάσεις δεδομένων. Η δυνατότητα να υπάρξει ταυτόχρονη πρόσβαση στα διαφορετικά format δεδομένων χωρίς μετατροπή κάνει τον MapServer μια από τις λίγες επιλογές για εκείνους που δεν μπορούν (ή δεν θέλουν) να κάνουν μια μετατροπή σε ένα συγκεκριμένο format.

5.2.1. Data Access and Performance

Ο MapServer υποστηρίζει αρκετά format. Μερικά format υπάρχουν στην εγκατάσταση του MapServer, ενώ τα άλλα προσεγγίζονται μέσω των βιβλιοθηκών GDAL/OGR. Η τελευταία προσέγγιση είναι απαραίτητη για format που δεν προγραμματίζονται άμεσα στον MapServer. Η πρόσβαση μέσω των βιβλιοθηκών προσθέτει ένα πρόσθετο επίπεδο επικοινωνίας μεταξύ του MapServer και της πηγής δεδομένων (που μπορούν να προκαλέσουν την κακή απόδοση σε μερικές περιπτώσεις).

Γενικά, τα format που υποστηρίζονται natively από τον MapServer τρέχουν γρηγορότερα από εκείνα που χρησιμοποιούν το GDAL/OGR. Παραδείγματος χάριν, το πιο βασικό format του MapServer είναι το ESRI shapefile ή GeoTiff. Το OGR υποστηρίζει το format αρχείων U.S. Census TIGER. Η διαφορά απόδοσης όταν φορτώνεται ένα TIGER αρχείο ή ένα shapefile μπορεί να είναι σημαντική. Εντούτοις, η χρησιμοποίηση του GDAL/OGR μπορεί να μην είναι πρόβλημα. Η περαιτέρω έρευνα δείχνει ότι τα format των δεδομένων είναι συχνά το πρόβλημα. Εάν τα δεδομένα σε ένα αρχείο είναι δομημένα με έναν τρόπο που δεν τα καθιστά εύκολα προσπελάσιμα ή απαιτεί πολυάριθμα επίπεδα μετάφρασης, επηρεάζει την ταχύτητα φόρτωσης των χαρτών.

Ο γενικός εμπειροτεχνικός κανόνας για την καλύτερη απόδοση είναι να χρησιμοποιηθεί το ESRI shapefile format ή το format εικόνας GeoTiff. Επειδή το gdal_translate και το ogrcgr μπορούν να γράψουν σε αυτά τα format, τα περισσότερα δεδομένα της πηγής μπορούν να μεταφραστούν χρησιμοποιώντας αυτά τα εργαλεία. Εάν υπάρχει πρόσβαση στα δεδομένα μέσω ενός δικτύου, η αποθήκευση των στοιχείων στη βάση δεδομένων PostGIS ίσως να είναι η καλύτερη επιλογή. Επειδή η PostGIS επεξεργάζεται τα ερωτήματα για τα δεδομένα στον server, μόνο τα επιθυμητά αποτελέσματα

στέλνονται από το δίκτυο. Η server-side επεξεργασία σε μια βάση δεδομένων PostGIS μπορεί να βελτιώσει σημαντικά την απόδοση των εφαρμογών του Mapserver.

5.2.2. Portability

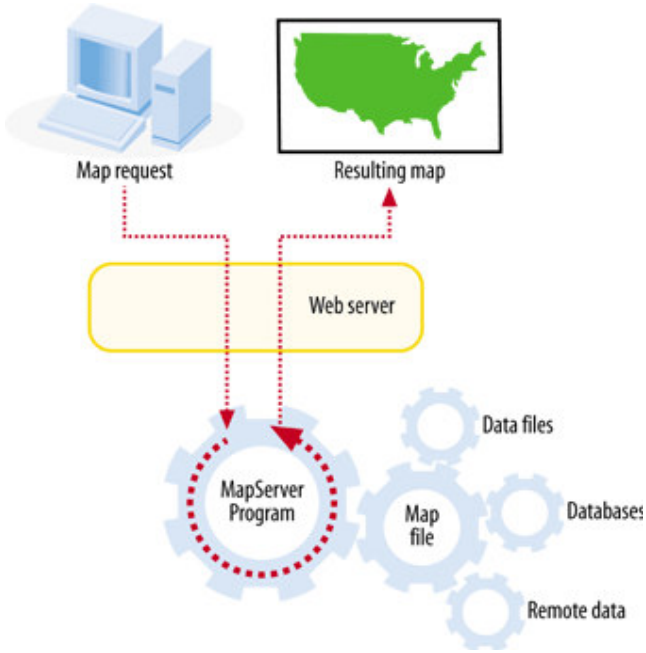
Ο MapServer και τα ενισχυτικά εργαλεία υποστήριξης του είναι διαθέσιμα για διαφορετικά λειτουργικά συστήματα. Επιπλέον, η λειτουργία του MapServer μπορεί να προσεγγιστεί από πολλές γλώσσες προγραμματισμού, καθιστώντας το πιθανό να ενσωματώσει τη λειτουργία του MapServer σε συνηθισμένα προγράμματα. Ο MapServer μπορεί να χρησιμοποιηθεί σε συνηθισμένα περιβάλλοντα όπου άλλοι web mapping servers ίσως να μην τρέχουν.

Επειδή ο MapServer είναι λογισμικό open source, οι προγραμματιστές μπορούν να βελτιώσουν, να διορθώσουν, να τροποποιήσουν τον πραγματικό κώδικα του MapServer, όπως επίσης και να τον προσαρμόσουν σε νέα λειτουργικά συστήματα ή πλατφόρμες. Στην πραγματικότητα, εάν χρειάζεται ένα νέο feature, ένας προγραμματιστής μπορεί να μισθωθεί για να το προσθέσει, και όλοι στην κοινότητα μπορούν να ωφεληθούν από αυτή την εργασία.

Ο MapServer είναι πρώτιστα μια εφαρμογή παρακολούθησης και δημιουργίας χαρτών. Οι χρήστες έχουν πρόσβαση στους χάρτες μέσω ενός web browser ή άλλων data sharing πρωτοκόλλων Διαδικτύου. Αυτό επιτρέπει την οπτική διανομή των πληροφοριών χαρτογράφησης και τη διανομή δεδομένων με άλλες εφαρμογές σε πραγματικό χρόνο χρησιμοποιώντας τις προδιαγραφές του OGC. Ο MapServer μπορεί να εκτελέσει μετατροπή δοδεδομένων διαβάζοντας διαφορετικά format και παρέχοντας πρόσβαση σε άλλους server ή εφαρμογές χρησιμοποιώντας κοινά πρωτόκολλα. Ο MapServer δεν είναι μόνο ένα εργαλείο ανάλυσης, αλλά μπορεί να παρουσιάσει πληροφορίες χαρτογράφησης χρησιμοποιώντας διαφορετικές χαρτογραφικές τεχνικές για να απεικονίσει τα αποτελέσματα.

5.3. Πως λειτουργούν οι εφαρμογές του Mapserver

Ο MapServer συνήθως λειτουργεί πίσω από μια web server εφαρμογή. Ο web server λαμβάνει τα αιτήματα για τους χάρτες και τα περνά στον MapServer για να τους δημιουργήσει. Ο MapServer παράγει τη ζητούμενη εικόνα χαρτών και την παραδίδει στο web server, ο οποίος την διαβιβάζει πίσω στο χρήστη.Η **Εικόνα 18** δείχνει πως ο χρήστης αλληλεπιδρά με το web server, ο οποίος στη συνέχεια, υποβάλλει τα αιτήματα στο MapServer πρόγραμμα.



Εικόνα 18: Βασική λειτουργία μιας Mapserver εφαρμογής

Η αρχική λειτουργία του MapServer είναι να διαβάζει τα δεδομένα από τις διάφορες πηγές και να τοποθετεί αυτά τα layers μαζί σε ένα γραφικό αρχείο, επίσης γνωστό ως *map image*. Κάθε layer «κάθεται» ή σχεδιάζεται πάνω από άλλα και τότε αποτυπώνεται σε ένα web-friendly γραφικό για να το δει ο τελικός χρήστης. Ένα καλό παράδειγμα των αποτελεσμάτων της διαδικασίας επικάλυψης και χαρτογράφησης μπορεί να φανεί στην **Εικόνα 19**. Φαίνεται μια δορυφορική εικόνα (από έναν μακρινό server), οι γραμμές των δρόμων και οι θέσεις των πόλεων. Οι ετικέτες των πόλεων παράγονται δυναμικά από τον MapServer.

Εικόνα 19: Χάρτης που απεικονίζει διάφορα layers πληροφορίας

Αυτή η διαδικασία της σχεδίασης (rendering) εμφανίζεται κάθε φορά που υποβάλλεται ένα αίτημα στον MapServer για να δημιουργήσει έναν νέο χάρτη, παραδείγματος χάριν, όταν ένας χρήστης μεγεθύνει ένα χάρτη (zoom in). Αυτή η διαδικασία εμφανίζεται επίσης όταν ένας χρήστης ζητά manually την επανασχεδίαση ενός χάρτη, όπως όταν αλλάξει το περιεχόμενο ενός layer δεδομένων, και ο χρήστης θέλει να δει την αλλαγή.

Ο MapServer μπορεί να λειτουργήσει με δύο διαφορετικούς τρόπους: το CGI και το MapScript mode. Στο CGI mode, ο MapServer λειτουργεί σε ένα web server περιβάλλον σαν CGI script. Σ' αυτή την περίπτωση είναι εύκολο να «στηθεί» και παράγει μια γρήγορη και απλή εφαρμογή. Στο MapScript mode, το MapServer API είναι προσπελάσιμο από την Perl, την Python, την PHP ή την Java. Η MapScript διεπαφή επιτρέπει μια ευέλικτη, πλούσια σε features εφαρμογή που έχει τη δυνατότητα να εκμεταλλευθεί τις template διευκολύνσεις του MapServer.

Ο Mapserver είναι ένα πρόγραμμα που βασίζεται σε templates. Όταν εκτελείται πρώτη φορά σαν απάντηση σε ένα web request, διαβάζει ένα configuration file (το οποίο αποκαλείται mapfile) που περιγράφει τα layers και τα άλλα συστατικά του χάρτη. Αφού διαβάσει το mapfile σχεδιάζει και αποθηκεύει το χάρτη. Στη συνέχεια, διαβάζει ένα ή περισσότερα HTML templates, τα οποία προσδιορίζονται στο mapfile. Κάθε template αποτελείται από συμβατικά HTML markup tags και τα MapServer substitution strings. Αυτά τα strings χρησιμοποιούνται, παραδείγματος χάριν, για να διευκρινίσουν τα μονοπάτια για την εικόνα του χάρτη, που ο MapServer έχει δημιουργήσει, για να προσδιορίσει ποια layers πρόκειται να αποδοθούν, και για να διευκρινίσουν το επίπεδο του zoom και την κατεύθυνση. Ο MapServer αντικαθιστά τις τρέχουσες τιμές για αυτά τα strings και στέλνει έπειτα το data stream στον web server, ο οποίος το διαβιβάζει έπειτα στον browser. Όταν ένας χρήστης που πραγματοποιεί ένα αίτημα αλλάζει οποιαδήποτε δεδομένα στη φόρμα της σελίδας (αλλάζοντας για παράδειγμα την κατεύθυνση του ζουμ ή το επίπεδο του ζουμ) και κάνει click στο κουμπί υποβολής (submit), MapServer λαμβάνει ένα αίτημα από τον web server με αυτές τις νέες τιμές. Κατόπιν ο κύκλος αρχίζει πάλι.

Ο MapServer εκτελεί αυτόματα διάφορες εργασίες κατά την παραγωγή ενός χάρτη. Ονομάζει τα features και αποτρέπει τις συγκρούσεις (collisions) μεταξύ των γειτονικών ετικετών. Επιτρέπει τη χρήση και bitmapped και TrueType γραμματοσειρών. Τα μεγέθη των ετικετών μπορούν να καθοριστούν ή να διαμορφωθούν στην ίδια κλίμακα με την κλίμακα του χάρτη. Η επιλογή να μην τυπωθούν οι ετικέτες για τις διευκρινισμένες σειρές κλίμακας χαρτών παρέχεται επίσης.

Επίσης ο MapServer δημιουργεί λεζάντες και scale bars (τα οποία μπορούν να διαμορφωθούν στο mapfile) και παράγει τους χάρτες αναφοράς. Ένας χάρτης αναφοράς παρουσιάζει το περιεχόμενο του χάρτη που απεικονίζεται εκείνη τη στιγμή. Παραδείγματος χάριν, εάν η περιοχή ενδιαφέροντος είναι η βόρεια Ντακότα, ο χάρτης αναφοράς θα παρουσίαζε ένα μικρό χάρτη της βόρειας Ντακότας, με τα περιεχόμενα του τρέχοντος χάρτη να αποτυπώνονται μέσα σε αυτόν. Το zoom in, zoom out και pan είναι υπό τον έλεγχο των χρηστών.

Ο MapServer δημιουργεί τους χάρτες τοποθετώντας τα layers το ένα πάνω από το άλλο. Κάθε ένα που αποδίδεται, τοποθετείται στην κορυφή του σωρού. Κάθε layer επιδεικνύει τα features που επιλέγονται από ένα σύνολο δεδομένων. Τα features που επιδεικνύονται μπορούν να επιλεγούν με τη χρησιμοποίηση κανονικών εκφράσεων Unix, συγκρίσεων strings και λογικών εκφράσεων. Λόγω της ομοιότητας των δεδομένων

και της ομοιότητας των παραμέτρων απεικόνισης (όπως κλίμακα, χρώματα, και ετικέτες), μπορείτε να σκεφτείτε ένα layer σαν ένα θέμα. Η απεικόνιση των layers είναι υπό interactive έλεγχο, που επιτρέπει στο χρήστη να επιλέξει ποια layers πρόκειται να αποδοθούν. Καθώς τα layers δεν μπορούν να παραχθούν on the fly, κενά layers μπορούν να γεμίσουν με δυναμικά δεδομένα και να χειριστούν μέσω URLs. Ο MapServer διαθέτει ισχυρές και περίπλοκες ικανότητες ερώτησης, αλλά στο CGI mode στερείται τα εργαλεία που επιτρέπουν το είδος ανάλυσης που παρέχεται από ένα GIS.

Αυτή η επισκόπηση έχει περιγράψει μερικά από τα χαρακτηριστικά γνωρίσματα του MapServer και έχει παρουσιάσει γιατί δεν είναι ένα full-featured GIS: δεν παρέχει κανένα ενσωματωμένο εργαλείο βάσης δεδομένων DBMS (database management system), οι αναλυτικές δυνατότητές της είναι περιορισμένες, και δεν έχει κανένα εργαλείο για georeferencing.

Δεδομένου ότι οι λειτουργίες του MapServer μπορούν να προσεγγιστούν μέσω ενός API από διάφορες γλώσσες προγραμματισμού (όπως η PHP, η Perl, η Java και η Python), μπορεί να χρησιμεύσει ως μια ισχυρή χωρική εφαρμογή που έχει πολλές από τις λειτουργίες ανάλυσης και αναφοράς ενός αληθινού GIS. Επιπλέον, ενώ δεν υπάρχει κανένα ενσωματωμένο εργαλείο για τον χειρισμό χωρικών δεδομένων, υπάρχουν σύνολα εργαλείων τρίτων που εκτελούν πολλές (αν και όχι όλες) από αυτές τις λειτουργίες.

Όταν ο MapServer τρέχει ως CGI σε ένα web περιβάλλον, μπορεί να αποδώσει χάρτες, παρουσιάζει τα δεδομένα των attributes, και να πραγματοποιήσει τις στοιχειώδεις χωρικές ερωτήσεις. Όταν προσεγγίζεται μέσω του API, η εφαρμογή γίνεται σημαντικά ισχυρότερη. Σε αυτό το περιβάλλον, ο MapServer μπορεί να εκτελέσει τις ίδιες εργασίες που θα μπορούσε να κάνει ως CGI, αλλά έχει επίσης πρόσβαση στις εξωτερικές βάσεις δεδομένων μέσω του προγράμματος ελέγχου, καθώς επίσης και την πιο σύνθετη λογική και ένα μεγαλύτερο ρεπερτόριο των πιθανών συμπεριφορών.

5.3.1. CGI Mapserver

Η απλούστερη μορφή του MapServer τρέχει σαν μία εκτελέσιμη CGI εφαρμογή σε έναν web server. Τεχνικά, ο MapServer θεωρείται μια HTTP-based stateless διαδικασία. Μια stateless διαδικασία σημαίνει ότι γίνεται η επεξεργασία ένα αιτήματος και μετά σταματά να τρέχει. Μια CGI εφαρμογή λαμβάνει τα αιτήματα από ένα web server, τα επεξεργάζεται, και μετά επιστρέφει μια απάντηση ή δεδομένα στο web server. Η CGI εφαρμογή είναι ο πιο δημοφιλής τρόπος χρησιμοποίησης του Mapserver και αυτό οφείλεται στην απλότητα του, που προκύπτει από το γεγονός ότι καμιά

προγραμματιστική εργασία δεν απαιτείται για να λειτουργήσει Οι ενέργειες που απαιτούνται για να τρέξει η εφαρμογή είναι να δημιουργηθεί ένα text-based, runtime configuration file, να δημιουργηθεί μια ιστοσελίδα, και μετά να στήνονται έτσι ώστε να εξυπηρετούνται από ένα web server.

Το CGI MapServer εκτελέσιμο επιδρά σαν ενδιάμεσο μεταξύ των χαρτογραφικών δεδομένων των αρχείων και του web server προγράμματος που ζητά το χάρτη. Τα αιτήματα περνούν υπό μορφή CGI παραμέτρων από τον web server στον MapServer. Οι εικόνες που δημιουργούνται από τον MapServer ανατροφοδοτούνται έπειτα στον web server και τελικά στον web browser του χρήστη.

Ο αρχικός στόχος είναι να παράγονται χάρτες σε ένα CGI περιβάλλον, στο οποίο ένας χρήστης έχει πρόσβαση σε έναν Apache web server από έναν web browser. Σε αυτό το περιβάλλον, ο Apache καλεί τον MapServer, περνώντας του τις μεταβλητές της φόρμας από τον web browser. Χρησιμοποιώντας αυτές τις πληροφορίες, ο MapServer παράγει τις εικόνες και μία ιστοσελίδα, την οποία ο Apache προωθεί πίσω στον browser. Φυσικά, ο MapServer χρειάζεται κάποιες παραμέτρους για να δημιουργήσει έναν χάρτη. Στην πραγματικότητα, μια CGI MapServer web εφαρμογή έχει τέσσερα συστατικά: το mapfile, μια HTML φόρμα αρχικοποίησης, ένα ή περισσότερα HTML template αρχεία, και μια χωρική βάση δεδομένων.

Ο MapServer, όπως όλες οι web εφαρμογές είναι βασισμένες σε ένα stateless πρωτόκολλο, πράγμα που σημαίνει ότι σε κάθε κλήση γνωρίζει μόνο ότι του έχει δηλώσει ο browser μόλις πριν. Η έλλειψη της προηγούμενης κατάστασης αποκλείει τη χρήση των εφαρμογών που πρέπει να κάνουν περισσότερα από το να απαντούν στην τελευταία ερώτηση που τους υποβλήθηκε. Εντούτοις, κάποια έξυπνη κωδικοποίηση μπορεί να προσφέρει ένα statefull server περιβάλλον και να δώσει στις web εφαρμογές τη δυνατότητα να εκτελέσουν πιο σύνθετες εργασίες.Παραδείγματος χάριν, η κατάσταση μπορεί να διατηρηθεί μεταξύ των κλήσεων με την αποθήκευση των πληροφοριών κατάστασης σε κρυμμένες μεταβλητές, στο URL, ή στα cookies. Απαιτείται όμως κάποια μέθοδος για την προσαρμογή της εφαρμογής έτσι ώστε να έχει τις πληροφορίες που απαιτεί στην πρώτη κλήση του Mapserver. Αυτό υλοποιείται από το αρχείο αρχικοποίησης. Σε μια CGI MapServer εφαρμογή, το αρχείο αρχικοποίησης είναι μια συμβατική HTML φόρμα με τις πληροφορίες έναρξης κωδικοποιημένες σε μεταβλητές μορφής. Σχεδόν οποιαδήποτε τιμή που χρησιμοποιεί ο MapServer μπορεί να μπει στο αρχείο έναρξης.

Όταν ο MapServer καλείται αρχικά από τον Apache από την HTML φόρμα αρχικοποίησης, μια μεταβλητή φόρμα χρησιμοποιείται για να διευκρινίσει το όνομα του mapfile (συνήθως με μια επέκταση .map). Στη συνέχεια διαβάζει το mapfile για να εντοπίσει τις γραμματοσειρές, τα σύμβολα, τα templates και τα χωρικά δεδομένα. Το mapfile διευκρινίζει επίσης το μέγεθος του προκύπτοντος χάρτη, τη γεωγραφική έκτασή του, και εάν είναι σε GIF, JPEG, ή PNG format. Έχοντας διαβάσει το mapfile, ο MapServer αποδίδει έπειτα μια ή περισσότερες εικόνες: τον ίδιο το χάρτη, τις λεζάντες και τη scalebar και ίσως και ένα χάρτη αναφοράς. Οι εικόνες αυτές αποθηκεύονται σε μια θέση, η οποία διευκρινίζεται στο mapfile.

Προκειμένου να παρουσιάσει τα αποτελέσματά του, ο MapServer πρέπει να μορφοποιήσει το χάρτη και τα σχετικά elements σαν μια ιστοσελίδα. Το ίδιο το πρόγραμμα δεν δημιουργεί την HTML. Αυτό που κάνει είναι να ανιχνεύει ένα HTML template για *τα substitution strings*. Τα substitution strings μπορούν να είναι αναφορές αρχείων, λεπτομέρειες της γεωμετρίας του χάρτη, προδιαγραφές των layers, ή παράγοντες ζουμ. Μπορούν επίσης να είναι οι τρέχουσες τιμές των CGI μεταβλητών όπως το μέγεθος εικόνας, το όνομα του mapfile, η έκταση του χάρτη, κλπ. Ο MapServer αντικαθιστά τα substitution strings με τις κατάλληλες τιμές και επιστρέφει την τροποποιημένη HTML σελίδα στον browser.

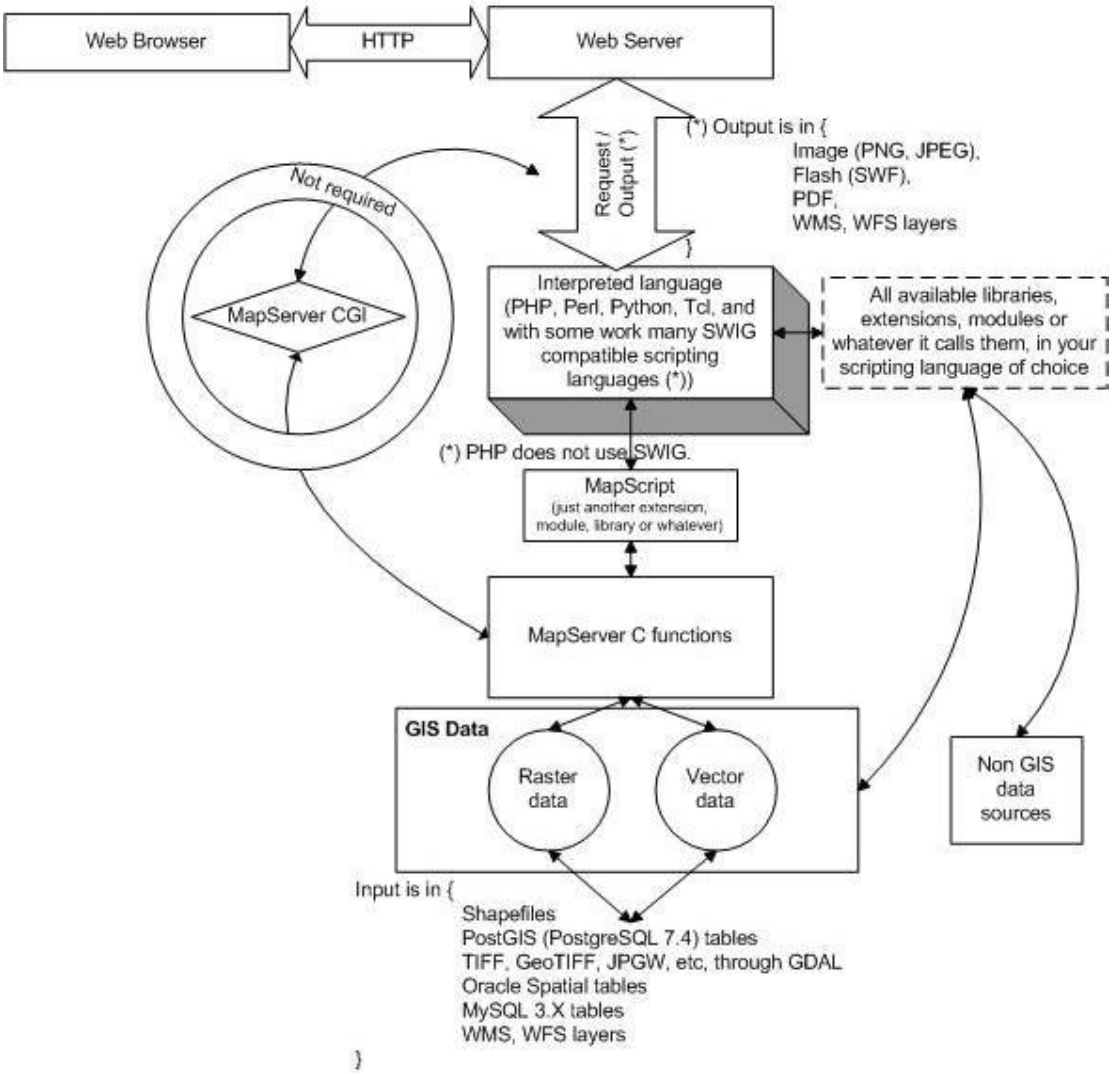
5.3.2. Mapscript

Όπως αναφέρθηκε παραπάνω ο CGI Mapserver μπορεί να κάνει αρκετά πράγματα αλλά έχει περιορισμένες δυνατότητες όσον αφορά τις απαντήσεις σε χωρικά ερωτήματα σε μια βάση δεδομένων. Υπάρχει, όμως και ένας άλλος πιο ισχυρός τρόπος χρήσης – κλήσης του Mapserver, η mapscript. Με τη mapscript δίνεται η δυνατότητα να πραγματοποιήσουμε πράγματα τα οποία δεν μπορούν να γίνουν μέσω του Mapserver CGI mode, όπως δυναμική δημιουργία layers, παραμετροποιήσιμη πλοήγηση, on the fly classification, προηγμένο έλεγχο των layers legends, καθώς και χωρικών αναζητήσεων. Η MapScript ενσωματώνει τη λειτουργικότητα του MapServer στις διάφορες scripting γλώσσες. Αυτό μειώνει το χρόνο προγραμματισμού για τους προγραμματιστές που θέλουν να προσθέσουν ικανότητες χαρτογράφησης σε μια εφαρμογή. Αντί ναδημιουργεί μια εξειδικευμένη μέθοδο για τη χαρτογράφηση, το MapScript API παρέχει μερικά ισχυρά, έτοιμα προς χρήση εργαλεία. Παρέχει επίσης έναν κατάλληλο τρόπο αλληλεπίδρασης με τα δεδομένα χαρτογράφησης μέσω της επιθυμητής από το χρήστη γλώσσα προγραμματισμού.

Η MapScript επιτρέπει στους χρήστες να φορτώσουν, να χειριστούν, και να δημιουργήσουν αρχεία χαρτών. Παραδείγματος χάριν, μπορεί κάποιος να αλλάξει τα settings των layers, να χειριστεί τις κλάσεις των mapfiles, να παράγει εικόνες εξόδου, να εξαγάγει τα χωρικά δεδομένα, κλπ. Επειδή χρησιμοποιεί κοινά scripting περιβάλλοντα, οι λειτουργίες της MapScript μπορούν να συνδυαστούν με άλλες λειτουργίες της script γλώσσας προγραμματισμού που επιθυμούμε.

Η MapScript υποστηρίζεται από διάφορες γλώσσες. Στην παρούσα φάση στις διαθέσιμες γλώσσες συγκαταλέγονται η PHP, η Python, η Perl, η Java, η C#, η TCL και η Ruby.

Η αρχιτεκτονική της Mapscript φαίνεται στην **Εικόνα 20**.



Εικόνα 20: Αρχιτεκτονική της Mapscript

Η Mapscript στηρίζεται στο αντικειμενοστραφή προγραμματισμό. Έτσι το API της είναι μια συλλογή από κλάσεις με μεθόδους και χαρακτηριστικά (attributes). Με δεδομένο ότι κάθε γλώσσα προγραμματισμού έχει διαφορετική δομή και συντακτικό το API της διαφοροποιείται κάπως ανάλογα με τη προγραμματιστική γλώσσα που χρησιμοποιούμε. Έτσι για παράδειγμα υπάρχουν οι Perl Mapscript, Python Mapscript, Java, TCL, Ruby και η PHP Mapscript. Στην εργασία μας χρησιμοποιήθηκε η PHP Mapscript [10].

5.3.2.1 PHP/Mapscript

Ο πηγαίος κώδικας της PHP/Mapscript περιέχεται μέσα στη διανομή του Mapserver αλλά δεν εγκαθίσταται με τον mapserver παρά μόνο εάν υπάρξει τέτοια απαίτηση από το χρήστη. Υπάρχουν περιπτώσεις που η mapscript δεν λειτουργεί καλά με την PHP 5. Αυτό οφείλεται σε δύο λόγους:

Ο πρώτος σχετίζεται με τον τρόπο που η PHP αλληλεπιδρά με τον Apache server. Η εξ' ορισμού εγκατάσταση της PHP δημιουργεί ένα εκτελέσιμο το οποίο προορίζεται να λειτουργεί σε ένα CGI περιβάλλον, δηλαδή εκτελείται κάθε φορά που καλείται ο Apache για την εκτέλεση ενός PHP script. Αντίθετα εάν η PHP μεταγλωττιστεί (compiled) ως ένα δυναμικά διαμοιραζόμενο αντικείμενο (Dynamic Shared Object – DSO) γίνεται τμήμα του Apache και δεν χρειάζεται να ξαναφορτώνεται με κάθε κλήση-αίτημα προς τον Apache. Η χρήση ως DSO είναι γρηγορότερη αλλά επειδή η Mapscript υποστηρίζεται μόνο ως CGI και όχι ως DSO ενδέχεται να προκληθούν προβλήματα. Για το λόγο αυτό θα εκτελούμε και την PHP σαν CGI.

Ο δεύτερος λόγος σχετίζεται με την ασυμβατότητα ανάμεσα στις ενσωματωμένες κανονικές εκφράσεις (regular expressions) της βιβλιοθήκης της PHP και της βιβλιοθήκης του συστήματος. Τόσο ο Apache όσο και η Mapscript χρησιμοποιούν τη βιβλιοθήκη συστήματος και επομένως ενδέχεται να δημιουργηθούν προβλήματα. Για αυτό μπορούμε είτε να κάνουμε recompile τον Apache και τη Mapscript χρησιμοποιώντας βιβλιοθήκες της PHP ή απλούστερα να προσδιορίσουμε ότι η PHP θα χρησιμοποιεί την βιβλιοθήκη του συστήματος. Παρακάτω περιγράφεται ο δεύτερος τρόπος.

5.3.2.2 Εγκατάσταση της PHP/Mapscript

Η ανάπτυξη της PHP/Mapscript όταν έχει ήδη εγκατασταθεί μια εφαρμογή CGI MapServer γίνεται με ένα αρκετά απλό recompile. Προκειμένου να αναπτυχθεί το PHP/MapScript module php_mapscript.so, πρέπει να δείξουμε τη θέση του directory εγκατάστασης της PHP. Αυτό γίνεται στο περιβάλλον ανάπτυξης με την επιλογή

διαμόρφωσης --with-php=/usr/local/include/php. Πρέπει να σημειωθεί ότι οι προεπιλεγμένες θέσεις των βιβλιοθηκών που χρησιμοποιούνται από την εντολή make μπορεί να μην είναι κατάλληλες για το περιβάλλον σας. Πάντως η εντολή configure απαιτεί να διευκρινιστεί ρητά η θέση του directory εγκατάστασης της PHP.

Ο MapServer δημιουργείται ξανά και η MapScript αναπτύσσεται με την εντολή:

make clean

./configure --with-proj --with-gdal --with-ogr --with-php=/usr/local/include/php

make

Αυτό θα δημιουργήσει το module php_mapscript.so μέσα στο ./mapscript/php3/. Αντιγράφοντας αυτό στα directories επεκτάσεων της PHP (στο περιβάλλον ανάπτυξης αυτό είναι /usr/lib/php/extensions/), εξασφαλίζεται ότι όλα τα directories που υπάρχουν στο μονοπάτι του module είναι αναγνώσιμα και εκτελέσιμα από τον Apache user. Η PHP πρέπει να ξέρει ότι πρέπει να φορτωθεί αυτό το module όταν ένα script εκτελείται. Για να συμβεί αυτό, πρέπει να παρεμβληθεί μια οδηγία στο php.ini αρχείο. Αυτή μπορεί να τοποθετηθεί μετά από την οδηγία extension_dir. Πρέπει να διαβάσει το extension = "php_mapscript.so".

5.4. Πακέτο εγκατάστασης Mapserver

Ο Mapserver μπορεί να εκτελείται σε μια πληθώρα από λειτουργικά συστήματα και ο τρόπος εγκατάστασής του ποικίλει ανάλογα με την αντίστοιχη πλατφόρμα [9].

5.4.1. Εγκατάσταση σε Windows

Ο πιο εύκολος τρόπος προκειμένου να χρησιμοποιήσουμε τον Mapserver στα Windows είναι να κατεβάσουμε τα τυποποιημένα πακέτα για Windows τα οποία περιλαμβάνουν οτιδήποτε χρειαζόμαστε για τον Mapserver [11].

Στη δικτυακή τοποθεσία http://maptools.org/ms4w/ υπάρχει το πιο εύκολο στη χρήση πακέτο εγκατάστασης, το MS4W (MapServer For Windows). Το βασικό πακέτο MS4W εγκαθιστά ένα προρυθμισμένο Web Server περιβάλλον που περιλαμβάνει τα ακόλουθα στοιχεία:

Apache HTTP Server version 2.2.8

PHP version 5.2.5

MapServer CGI 5.0.2

MapScript 5.0.2 (CSharp, Java, PHP, Python)

Περιλαμβάνει υποστήριξη για την Oracle 10g, και SDE data

MrSID support built-in

GDAL/OGR 1.5.0 και Utilities

MapServer Utilities

PROJ Utilities

Shapelib Utilities

Shp2tile Utility

Shpdiff Utility

AVCE00 Utilities

OGR/PHP Extension 1.0.0

OWTChart 1.2.0

DEMtools Utilities

Το πακέτο αυτό είναι ένα zip αρχείο το οποίο απλά αποσυμπιέζουμε στο root του σκληρού (C:\). Δημιουργείται έτσι ένα πλήθος υποφακέλων κάτω από το φάκελο C:\ms4w\, συμπεριλαμβανομένου του φακέλου Apache όπου βρίσκεται το αντίστοιχος server. Στη συνέχεια εκτελούμε το αρχείο apache-install.bat. Ο Apache τρέχει ως service και ξεκινά σε κάθε εκκίνηση του υπολογιστή. Για να ελέγξουμε τη λειτουργία του server καλούμε στον browser http://localhost/ ή http://127.0.0.1/ οπότε εμφανίζεται η welcome screen του MS4W.

Προκειμένου να μπορέσουμε να εκτελέσουμε την php/mapscript θα πρέπει να τροποποιήσουμε το αρχείο php.ini ώστε να φορτώνεται η αντίστοιχη βιβλιοθήκη. Έτσι στο php.ini προσθέτουμε τη γραμμή extension=php_mapscript.dll. Στη συνέχεια εκτελούμε το αρχείο apache-restart.bat.

5.5. Mapfile

Το mapfile αθορίζει πώς θα φαίνεται ο χάρτης, ποιο θα είναι το μέγεθος του, και ποια layers θα εμφανίζονται. Υπάρχουν πολλές διαφορετικές προσεγγίσεις διαμόρφωσης του mapfile. Τα έγγραφα αναφοράς του mapfile που βρίσκονται στο web site του MapServer είναι μια απαραίτητη αναφορά για την εκμάθηση των νέων features. Μπορείτε να βρείτε τα έγγραφα αυτά στη διεύθυνση http://mapserver.gis.umn.edu/doc/mapfile-reference.html.

Τα mapfiles είναι απλά αρχεία κειμένου που χρησιμοποιούν μια ιεραρχική, ή nested, δομή αντικειμένων για να καθορίσουν τα χαρακτηριστικά για το χάρτη. Το mapfile αποτελείται από αντικείμενα. Κάθε αντικείμενο έχει μια λέξη κλειδί για να αρχίζει και τη λέξη END για να κλείνει. Τα αντικείμενα μπορούν να έχουν μέσα τους άλλα αντικείμενα, όπου αυτό χρειάζεται. Πιο συγκεκριμένα, ένα layer θα περιλαμβάνει αντικείμενα που καθορίζουν πως το αντικείμενο layer θα σχεδιαστεί, παραδείγματος χάριν πώς θα δημιουργούνται οι ετικέτες για αυτό το layer. Η **Εικόνα 21** δείχνει ένα απλό mapfile, που χρησιμοποιεί μόνο 15 γραμμές κειμένου.

```
MAP                         # Start of MAP object
  SIZE 600 300
  EXTENT -180 -90 180 90
  LAYER                     # Start of LAYER object
    NAME countries
    TYPE POLYGON
    STATUS DEFAULT
    DATA countries_simpl
    CLASS                   # Start of CLASS object
      STYLE                 # Start of STYLE object
        OUTLINECOLOR 100 100 100
      END                   # End of STYLE object
    END                     # End of CLASS object
  END                       # End of LAYER object
END                         # End of MAP object and map file
```

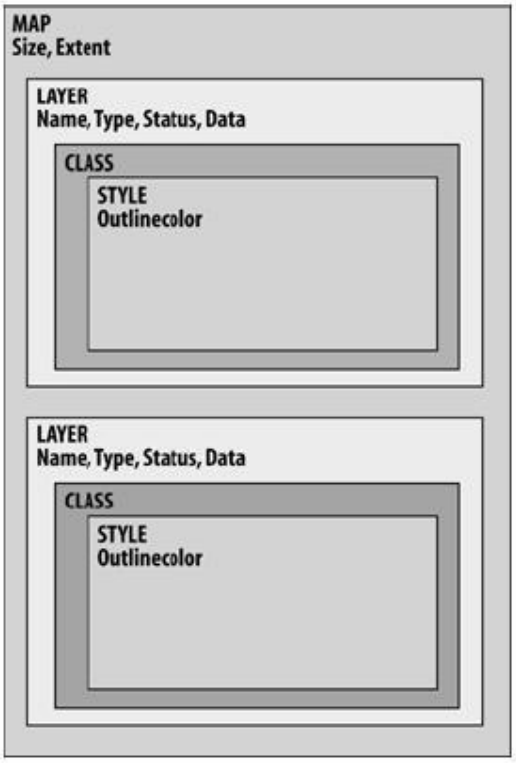
Εικόνα 21: Ένα απλό mapfile με ένα layer

Υπάρχουν μόνο τέσσερα αντικείμενα σε αυτό το mapfile:

- το MAP αντικείμενο, που καθορίζει τα χαρακτηριστικά του χάρτη για όλο το αρχείο και μέσα στο οποίο περιλαμβάνονται όλα τα υπόλοιπα αντικείμενα
- το LAYER αντικείμενο, που καθορίζει το layer που θα σχεδιάσει στο χάρτη
- το CLASS αντικείμενο, που καθορίζει τις κλάσεις των χαρακτηριστικών για το layer

- το STYLE αντικείμενο, το οποίο καθορίζει πώς τα features θα σχεδιαστούν στην κλάση

Τα υπόλοιπα αντικείμενα, όπως και τα παραπάνω, θα αναλυθούν στη συνέχεια. Η Εικόνα 22 δείχνει τη δομή ενός mapfile.



Εικόνα 22: Δομή ενός mapfile

Ακολουθεί ανάλυση των κυριότερων αντικειμένων και των features του κάθε αντικειμένου του mapfile.

5.5.1. Map Object

Το map object δεν ορίζεται απλά μέσα στο mapfile - *είναι* το mapfile [10]. Είναι ο πατέρας όλων των αντικειμένων του mapfile και καθορίζει τα χαρακτηριστικά της εφαρμογής που είναι global.

FONTSET [filename]

Καθορίζει το μονοπάτι (απόλυτο ή σχετικό με τη θέση του mapfile) στο αρχείο που ορίζει τη χαρτογράφηση από τα αρχεία aliases γραμματοσειρών στα αρχεία γραμματοσειρών True Type.

IMAGECOLOR [int r][int g][int b]

Καθορίζει το χρώμα υποβάθρου της εικόνας του χάρτη. Αυτό το χρώμα γίνεται διαφανές εάν το TRANSPARENT έχει επιλεγεί. Οι τιμές είναι ακέραιοι αριθμοί ενός byte στη σειρά 0 έως 255, αναπαριστώντας σχετικά ποσά κόκκινου, πράσινου, και μπλε.

IMAGETYPE [gif | png | jpeg | wbmp | gtiff | swf | userdefined]

Καθορίζει τη μορφή της εικόνας εξόδου. Ο τύπος εικόνας μπορεί να είναι ένας από τους OUTPUTFORMAT τύπους που αναγνωρίζονται από τον MapServer (που περιγράφονται αργότερα σ' αυτή την ενότητα) ή αυτό μπορεί να είναι ένας καθορισμένος από το χρήστη τύπος προσδιορισμένος από το όνομά του όπως ορίζεται από τη λέξη κλειδί NAME στην κατάλληλη OUTPUTFORMAT δήλωση.

LEGEND

Δείχνει την έναρξη ενός LEGEND αντικειμένου.

NAME [name]

Καθορίζει το όνομα που χρησιμοποιείται για να προσδιορίσει το χάρτη εξόδου. Ένας αριθμός ID (που παράγεται από σύνδεση της ταυτότητας χρόνου και της διαδικασίας ID) επισυνάπτεται σε αυτό το όνομα για να παρέχει μια μοναδική ID ταυτότητα.

SHAPEPATH [path]

Καθορίζει το directory στο οποίο είναι αποθηκευμένα τα shapefiles. Η τιμή που ορίζεται στο SHAPEPATH προτάσσεται στο σύνολο των δεδομένων που καθορίζεται από το keyword του layer DATA.

SIZE [int x][int y]

Καθορίζει το πλάτος και το ύψος της εικόνας του χάρτη σε pixels.

STATUS [on | off]

Default: on

Καθορίζει εάν η εικόνα του χάρτη δημιουργείται.

UNITS [feet | inches | kilometers | meters | miles | dd]

Default: n/a

Καθορίζει τις μονάδες μέτρησης των αποστάσεων στο χάρτη. Αυτή η λέξη κλειδί επηρεάζει τους υπολογισμούς της κλίμακας και τις scale bars. Το dd δείχνει τους δεκαδικούς βαθμούς.

EXTENT [minx][miny][maxx][maxy]

Default: n/a

Καθορίζει την έκταση του χάρτη. Το extent toy χάρτη καθορίζεται από τις συντεταγμένες του κάτω αριστερά σημείου του χάρτη (minx, miny) και του πάνω δεξιά σημείου (maxx, maxy). άθη στον καθορισμό του extent μπορούν να οδηγήσουν σε κενούς χάρτες ή σε διαστρεβλωμένους χάρτες. Το extent πρέπει να καθοριστεί, και πρέπει να είναι στο ίδιο σύστημα συντεταγμένων με το αντικείμενο PROJECTION του χάρτη (εάν υπάρχει). Υπάρχει η δυνατότητα να καθοριστεί από τον χρήστη και αν αυτό δεν γίνει τότε θα ορίσει κάποιο ο mapserver.

5.5.2. Layer Object

Τα στοιχεία του LAYER αντικειμένου καθορίζουν τα χωρικά δεδομένα που πρόκειται να μεταδοθούν και πώς αυτά πρόκειται να ταξινομηθούν. Αρχίζει με τη λέξη κλειδί LAYER και ολοκληρώνεται με τη λέξη κλειδί END. Τα layers σχεδιάζονται στη σειρά που βρίσκονται στο mapfile, και τα επόμενα στρώματα δίνονται πάνω από εκείνα που αποδόθηκαν πρώτα.

CLASSITEM [attribute]

Default: none

Καθορίζει το όνομα των attributes που χρησιμοποιούνται για να ταξινομήσουν τα features όταν η λέξη κλειδί EXPRESSION χρησιμοποιείται για να εκτελέσει κανονικές συγκρίσεις ή συγκρίσεις σε strings

LABELITEM [attribute]

Default: n/a

Καθορίζει το όνομα των attributes των οποίων η τιμή χρησιμοποιείται για να ονομάσει ένα feature.

NAME [string]

Default: none

Καθορίζει το όνομα του layer (μέγιστο 20 χαρακτήρες). Αυτό το όνομα χρησιμοποιείται ως τιμή του layer της φόρμας της CGI μεταβλητής για να επιτρέψει στο layer να είναι on και off αμφίδρομα.

SIZEUNITS [pixels | feet | inches | kilometers | meters | miles]

Default: pixels

Θέτει τις μονάδες του CLASS αντικειμένου SIZE, το οποίο καθορίζει το μέγεθος των συμβόλων που χρησιμοποιούνται για να για να σχεδιάσουν τα features.

STATUS [on | off | default]

Default: n/a

Καθορίζει εάν ένα layer θα εμφανίζεται και εάν μπορεί να επιλεχτεί στη θέση on ή στη θέση off. Η επιλογή on καθορίζει ότι ένα layer θα εμφανίζεται πάντα, η επιλογή default καθορίζει ότι ένα layer θα εμφανίζεται, αλλά μπορούμε κάποια στιγμή να επιλέξουμε να μην εμφανίζεται ενώ η επιλογή off καθορίζει ότι ένα layer δεν θα εμφανίζεται, αλλά μπορούμε και κάποια στιγμή να επιλέξουμε να εμφανίζεται.

5.5.3. Class Object

Το CLASS αντικείμενο καθορίζει τις ιδιότητες εμφάνισης των features. Κάθε layer πρέπει να περιέχει μια ή περισσότερες κλάσεις. Ένα CLASS αντικείμενο εισάγεται με τη λέξη κλειδί CLASS και ολοκληρώνεται με τη λέξη κλειδί END.

COLOR [int r][int g][int b]

Default: 0 0 0

Καθορίζει το χρώμα που θα χρησιμοποιηθεί για να δώσει στα features. Οι τιμές είναι ακέραιοι αριθμοί 1-byte από 0 έως 255, αντιπροσωπεύοντας σχετικά ποσά κόκκινου, πράσινου, και μπλε.

EXPRESSION [string]

Default: n/a

Καθορίζει την έκφραση που αξιολογεί για να καθορίσει εάν ένα feature πρέπει να περιληφθεί στην κλάση. Η τιμή που ορίζεται στο EXPRESSION αντιπροσωπεύει έναν από τους ακόλουθους τύπους εκφράσεων: μια σύγκριση string, μια κανονική έκφραση, μια λογική έκφραση, ή μια string λειτουργία . Μια σύγκριση string ταιριάζει με το περιεχόμενο των attributes που προσδιορίζονται από τη λέξη κλειδί CLASSITEM με ένα string. Η σύγκριση αυτή είναι case sensitive. Αν οι δύο εκφράσεις ταιριάζουν, το feature συμπεριλαμβάνεται στην κλάση. Εάν το string περιέχει κενά ή τους ειδικούς χαρακτήρες (όπως tabs ή τις νέες γραμμές), πρέπει να μπει σε αποσιωπητικά. Μια κανονική έκφραση ταιριάζει με το περιεχόμενο tattribute που προσδιορίζεται από τη λέξη κλειδί CLASSITEM με την κανονική έκφραση που διευκρινίζεται από το string. Η κανονική έκφραση πρέπει να οριοθετηθεί από κάθετους, (π.χ., / κανονική έκφραση/). Μια λογική έκφραση αποτελείται από παρενθέσεις, στις οποίες υπάρχει ένας συνδυασμός

ονομάτων των attributes (που οριοθετούνται από αγκύλες), οι Boolean operators AND και OR, οι αριθμητικοί operators σύγκρισης =, >, <, <=, >=, και !=, οι operators σύγκρισης string eq. lt, gt, le, ge, eq, and ne και τιμές σύγκρισης. Παραδείγματος χάριν, EXPRESSION ([area]>100) θα επιλέξει τα features με area *μεγαλύτερη* από 100, και EXPRESSION ([status]!=0) θα επιλέξει τα αρχεία με τη κατάσταση *μη ίση* με 0. Τα attributes, των οποίων η τιμή είναι string και οι τιμές σύγκρισης πρέπει να μπουν σε αποσιωπητικά. Παραδείγματος χάριν, η EXPRESSION ('[type]' ne 'river') θα επιλέξει μόνο εκείνα τα features με type μη ίσο με river. Αυτήν την περίοδο, η μόνη string λειτουργία που υποστηρίζεται από τον MapServer είναι το length(), το οποίο υπολογίζει το μήκος ενός attribute, του οποίου η τιμή είναι string. Η EXPRESSION (length('[name]") < 2) επιλέγει τα features με σύντομα ονόματα.

MAXSIZE [int N]

Default: 50

Καθορίζει το μέγιστο μέγεθος (σε pixels) στο οποίο ένα σύμβολο θα σχεδιασθεί.

MINSCALE [double N]

Default: n/a

Καθορίζει την ελάχιστη κλίμακα.

MINSIZE [int N]

Default: 0

Καθορίζει το ελάχιστο μέγεθος (σε pixels) στο οποίο ένα σύμβολο θα σχεδιασθεί.

NAME [string]

Default: n/a

Καθορίζει το όνομα για την κλάση για χρήση στη λεζάντα. Εάν ένα όνομα δεν καθορίζεται, το feature θα σχεδιασθεί, αλλά δεν θα φαίνεται στη λεζάντα.

OUTLINECOLOR [int r][int g][int b]

Default: n/a

Καθορίζει το χρώμα περιγράμματος (για τα πολύγωνα μόνο). Οι τιμές είναι ακέραιοι αριθμοί ενός byte από 0 έως 255, αντιπροσωπεύοντας σχετικά ποσά κόκκινου, πράσινου, και μπλε.

MAXSIZE [int N]

Default: 50

Καθορίζει το μέγιστο μέγεθος (σε pixels) στο οποίο ένα σύμβολο θα σχεδιασθεί.

MINSIZE [int N]

Default: 0

Διευκρινίζει το ελάχιστο μέγεθος (σε pixels) στο οποίο ένα σύμβολο θα σχεδιασθεί.

SIZE [int N]

Καθορίζει το ύψος ενός συμβόλου (σε pixels).

5.5.4. Outputformat Object

Το OUTPUTFORMAT αντικείμενο χρησιμοποιείται για να καθορίσει το format του τελικού αρχείου που θα δώσει ο Mapserver. Αρχίζει με τη λέξη κλειδί OUTPUTFORMAT και ολοκληρώνεται με τη λέξη κλειδί END. Επεκτείνει την έννοια του format της εικόνας (όπως το GIF ή JPEG) για να περιλάβει λεπτομέρειες για τη δομή της εικόνας, του χρώματος και ποιοι drivers γραφικών πρέπει να χρησιμοποιηθούν για να παραχθεί μια εικόνα.

DRIVER ["name"]

Default: n/a

Καθορίζει το όνομα του driver γραφικών που χρησιμοποιείται για να παράγει την εικόνα. Οι παρακάτω GD drivers υποστηρίζονται: "GD/Gif", "GD/PNG", "GD/WBMP", and "GD/JPEG". Το όνομα του flash driver είναι "SWF". Για την έξοδο που υποστηρίζεται από το GDAL, το όνομα αποτελείται από ένα string "GDAL" συνοδευόμενο με το GDAL shortname για το format του αρχείου εξόδου, π.χ. "GDAL/GTiff".

MIMETYPE [mimetype]

Default: n/a

Καθορίζει τον τύπο mime που χρησιμοποιείται για το αποτέλεσμα.

NAME [name]

Default: n/a

Καθορίζει το όνομα που χρησιμοποιείται από τη λέξη κλειδί του mapfile IMAGETYPE για να παραπέμψει στο format που θα αποδώσει ο mapserver.

Label Object

Το Label αντικείμενο καθορίζει ένα κείμενο ή ένα σύμβολο που χρησιμοποιείται για να ονομάσει ένα feature. Αρχίζει με τη λέξη κλειδί LABEL και ολοκληρώνεται με τη λέξη κλειδί END.

ANGLE [auto | double N]

Default: n/a

Καθορίζει τη γωνία σχεδίασης μια ετικέτας. Μια γωνία 0 θα προκαλέσει τη σχεδίαση μιας ετικέτας παράλληλης με το κάτω μέρος του χάρτη. Για τα layers γραμμών μόνο, η τιμή auto προκαλεί τη σχεδίαση μιας ετικέτας ευθυγραμμισμένης με το feature.

COLOR [int r][int g][int b]

Default: 0 0 0

Καθορίζει το χρώμα που θα χρησιμοποιηθεί για να αποδώσει το κείμενο των ετικετών. Οι τιμές είναι ακέραιοι αριθμοί ενός byte από 0 έως 255, αντιπροσωπεύοντας τα σχετικά ποσά κόκκινου, πράσινου, και μπλε.

FONT [name]

Default: none

Διευκρινίζει το ψευδώνυμο της γραμματοσειράς που χρησιμοποιείται για να ονομάσει ένα feature. Ο MapServer μεταφράζει το ψευδώνυμο στο μονοπάτι που βρίσκεται στο αρχείο που καθορίζεται από τη λέξη κλειδί FONTSET.

OUTLINECOLOR [int r][int g][int b]

Default: n/a

Καθορίζει το χρώμα που χρησιμοποιείται για να δημιουργήσει ένα περίγραμμα εύρους 1 pixel γύρω από το κείμενο της ετικέτας. Οι τιμές είναι ακέραιοι αριθμοί ενός byte από 0 έως 255, αντιπροσωπεύοντας τα σχετικά ποσά κόκκινου, πράσινου, και μπλε.

SIZE [int N] | [tiny | small | medium | large | giant]

Default: n/a

Καθορίζει το μέγεθος του κειμένου της ετικέτας. Το μέγεθος των ετικετών True Type καθορίζεται σε pixels με μια ακέραια τιμή.Το μέγεθος μιας bitmapped ετικέτας ορίζεται με μια τιμή tiny, small, medium, large ή giant.

Ο mapserver εκτός από ετικέτες μπορεί να σχεδιάσει με τον ίδιο τρόπο λεζάντες, χάρτες αναφοράς, scalebars κτλ. Για κάθε ένα από τα παραπάνω που θα χρειαστεί να

σχεδιάσει ο mapserver υπάρχουν και ξεχωριστά objects, π.χ. Legend Object, Reference Map Object, Scalebar Object, με παρόμοια λειτουργία με αυτή του Label object και παρόμοια features με κάποιες μικρές διαφοροποιήσεις. Οπότε κρίνεται σκόπιμο να μην γίνει εκτενέστερη αναφορά σε αυτά.

5.5.5. WEB Object

Το WEB αντικείμενο καθορίζει τη web διεπαφή, συμπεριλαμβανομένων των μονοπατιών, των URLs, των template αρχείων, και άλλες λεπτομέρειες που επηρεάζουν τον τρόπο που η εφαρμογή αποκρίνεται στο χρήστη. Οι ιδιότητες ενός Web αντικειμένου είναι οι ακόλουθες:

HEADER [filename]

Default: n/a

Καθορίζει το όνομα του HTML template αρχείου που εμφανίζεται πριν την παρουσίαση οποιοδήποτε αποτελεσμάτων ερωτήσεων. Χρησιμοποιείται για τρόπους ερώτησης που δίνουν πολλαπλά αποτελέσματα.

IMAGEPATH [path]

Default: n/a

Καθορίζει το μονοπάτι στο directory όπου οι εικόνες και τα προσωρινά αρχεία αποθηκεύονται. Το μονοπάτι πρέπει να τελειώνει με ένα / ή \ (ανάλογα με την πλατφόρμα).

IMAGEURL [path]

Default: n/a

Καθορίζει το URL που δείχνει στο directory που οι εικόνες αποθηκεύονται. Ο browser χρησιμοποιεί αυτό το μονοπάτι για να ανακτήσει τις εικόνες.

TEMPLATE [filename | url]

Default: n/a

Καθορίζει το κύριο HTML template αρχείο που χρησιμοποιείται για να εμφανίσει ένα χάρτη με αμφίδρομο τρόπο.

ΚΕΦΑΛΑΙΟ 6

ΥΛΟΠΟΙΗΣΗ ΕΦΑΡΜΟΓΗΣ

6.1. Γενικά

Η εφαρμογή που δημιουργήσαμε στηρίζεται στο μοντέλο server-client και κάνει χρήση του mapserver, ο οποίος, για την λειτουργία του ως web browser, υλοποιεί τον Apache server. Η εφαρμογή αποτελείται από δύο κύρια τμήματα. Το ένα υλοποιείται στον server και το άλλο στον client. Στην πλευρά του server αναπτύχθηκε κώδικας σε php/mapscript ο οποίος σε συνδυασμό με χρήση ενός αρχείου mapfile δημιουργεί τις απαιτούμενες διανυσματικές εικόνες χαρτών τύπου svg που τελικά αποστέλλονται στον χρήστη. Τα αρχεία που σημιουργούνται και αποστέλλονται αφορούν το ισόγειο κτήριο του Πανεπιστημίου Αθηνών / Τμήματος Πληροφορικής. Χρησιμοποιήσαμε ως χάρτη αναφοράς την κάτοψη του κτηρίου αυτού διαστάσεων 86m x 37m. Στην πλευρά του client ουσιαστικά είναι το interface του χρήστη με τον server. Δηλαδή είναι το κομμάτι της εφαρμογής όπου γίνεται η απεικόνιση των χαρτών που έχει αποστείλει ο server και περιλαμβάνει και το διαδραστικό κομμάτι με το χρήστη (π.χ. επιλογή zoom, pan, layers, κλπ). Επισημαίνεται ότι το υπόψη interface υλοποιήθηκε με χρήση του προτύπου svg και δεν είναι απλά μια html σελίδα. Η επικοινωνία του server με τον client γίνεται μέσω ασύγχρονων κλήσεων (τεχνολογία Asynchronous Javascript and XML – AJAX γεγονός που επιτρέπει την παρουσίαση των αποτελεσμάτων των αιτημάτων απεικόνισης χαρτών με έναν πολύ αποτελεσματικό και εργονομικό τρόπο. Τα πλεονεκτήματα της χρήσης Ajax οδήγησαν στη χρήση αυτής της τεχνολογίας από πολύ γνωστές διαδικτυακές εφαρμογές με πιο χαρακτηριστική αυτή του Gmail.

Γενικά για την υλοποίηση της εφαρμογής απαιτήθηκε πέραν από τη μελέτη του τρόπου λειτουργίας του mapserver, η εξοικείωση και χρήση των διαδικτυακών τεχνολογιών php, mapscript, javascript, Ajax, xml, DOM) [12] [13] [14] [15] [16] [17].

6.2. Τμήμα εφαρμογής στον Server

Το κομμάτι της εφαρμογής που υλοποιείται στον server αφορά τη δημιουργία των αρχείων «UniversityFinal.map» και «getUniversityMap.php». Το πρώτο είναι το αρχείο mapfile που σχετίζεται με τη μορφή απεικόνισης των χαρτών, ενώ το δεύτερο δημιουργεί, κάνοντας χρήση php/mapscript, το χάρτη που έχει αιτηθεί ο χρήστης. Επισημαίνεται ότι θα μπορούσαμε να φτιάξουμε το ζητούμενο χάρτη χωρίς τη χρήση του mapfile αρχείου χρησιμοποιώντας μόνο php/mapscript κώδικα, αλλά προτιμήθηκε ο

διαχωρισμός του στατικού κομματιού που σχετίζεται με τον τρόπο απεικόνισης των χαρτών και του δυναμικού που έχει να κάνει με το τι θα απεικονιστεί.

Στις περιπτώσεις που δημιουργούμε ένα map αντικείμενο με τη mapscript χρησιμοποιώντας (διαβάζοντας) ένα mapfile, όλες οι παράμετροι του χάρτη, τα επίπεδα (layers), οι κλάσεις (classes) και οι ιδιότητες (attributes) του mapfile μεταφράζονται σε mapscript αντικείμενα. Για τις ιδιότητες που δεν έχουν προσδιοριστεί επιλέγονται αυτόματα οι εξ ορισμού τιμές και εφόσον στο mapfile προσδιορίζεται ο τύπος του χάρτη (svg, jpg, κλπ) η mapscript θα παράγει τον τύπο αυτό. Επιπρόσθετα, βέβαια, όλα τα αντικείμενα της mapscript μπορούν να τροποποιηθούν προγραμματιστικά, κάτι που δε μπορεί να γίνει στο mapfile. Με τη χρήση του αρχείου mapfile, μπορούμε να επικεντρωθούμε στις δυνατότητες που παρέχουν οι συναρτήσεις της mapscript για τη δυναμική δημιουργία του χάρτη, χωρίς να ασχολούμαστε με τις λεπτομέρειες του τρόπου απεικόνισης, καθώς αυτές καθορίζονται στο mapfile.

6.2.1. Σύστημα συντεταγμένων χάρτη

Για να αναπαρασταθεί με πιστότητα η πραγματική μορφή της επιφάνειας της γης στο χαρτί απαιτούνται σύνθετοι μαθηματικοί υπολογισμοί. Ακόμα και αν η γη ήταν μια τέλεια σφαίρα, η προβολή της επιφάνειάς της σε ένα επίπεδο δεν θα ήταν απλή υπόθεση. Ωστόσο, η επιφάνεια της γης είναι ακανόνιστη και δεν μπορεί να παρασταθεί από κάποιο γεωμετρικό σχήμα. Έτσι, για τους μαθηματικούς υπολογισμούς που απαιτούνται στη δημιουργία των υποβάθρων των χαρτών χρησιμοποιείται ένα θεωρητικό (γεωμετρικό) σχήμα, το ελλειψοειδές (το σχήμα που προκύπτει από την περιστροφή μιας έλλειψης γύρω από τον άξονα των πόλων). Τα γεωμετρικά στοιχεία του ελλειψοειδούς επιλέγονται έτσι ώστε η επιφάνεια που προκύπτει να προσεγγίζει το γεωειδές, δηλαδή μια επίσης θεωρητική επιφάνεια που ταυτίζεται με το μέσο επίπεδο της θαλάσσιας επιφάνειας και τη θεωρητική προέκτασή της κάτω από τις ηπείρους και η οποία προσδιορίζεται με μετρήσεις του πεδίου βαρύτητας. Το γεωειδές είναι μια "πραγματική" επιφάνεια αναφοράς, στο βαθμό που προσεγγίζει αδρά τη μορφή της επιφάνειας της γης.

Επειδή η μετατροπή μιας σφαιρικής επιφάνειας (όπως η γη) σε επίπεδη δεν μπορεί να γίνει χωρίς παραμορφώσεις, έχουν δημιουργηθεί διάφορα συστήματα προβολής, από τα οποία, άλλα διατηρούν τις αναλογίες των εμβαδών, άλλα των μηκών και άλλα τις γωνίες ανάμεσα στο χάρτη και το πεδίο. Το προβολικό σύστημα το οποίο έχει καθιερωθεί και χρησιμοποιείται διεθνώς (εκτός από τις πολικές περιοχές) είναι η

Παγκόσμια Εγκάρσια Μερκατορική των 6 μοιρών, πιο γνωστή ως UTM (Universal Transverse Mercator).

Ένα γεωδαιτικό σύστημα αναφοράς ορίζεται με την επιλογή ενός ελλειψοειδούς (που προσεγγίζει όσο το δυνατόν καλύτερα το γεωειδές) και ενός προβολικού συστήματος για την απεικόνιση της επιφάνειας του ελλειψοειδούς στο επίπεδο. Ο mapserver έχει τη δυνατότητα να χρησιμοποιήσει διάφορα γεωδαιτικά συστήματα αναφοράς (π.χ. ED50, WG84 κλπ).

Με δεδομένο ότι στην εργασία μας αναφερόμαστε σε μικρής έκτασης χάρτη (χάρτης κτηρίου) είναι σαφές ότι δεν έχει νόημα η χρήση γεωδαιτικών συστημάτων. Από την άλλη βέβαια είναι απαραίτητο να υπάρχει ένα σύστημα συντεταγμένων μοναδικά ορισμένο προκειμένου να δίνεται η δυνατότητα πλοήγησης στο χάρτη. Για το σκοπό αυτό ορίστηκε ένα σύστημα συντεταγμένων (x,y) με αρχή την κάτω αριστερή γωνία του χάρτη. Όλα τα άλλα σημεία είναι ορισμένα με βάση την απόσταση σε μέτρα από την αρχή των αξόνων. Οι αποστάσεις αυξάνονται κινούμενοι προς τα πάνω και δεξιά. Έτσι το πιο απομακρυσμένο σημείο του χάρτη είναι η πάνω δεξιά γωνία του η οποία έχει συντεταγμένες (86,37). Επισημαίνεται ότι οι αποστάσεις αυτές είναι πραγματικές με βάση τα σχέδια της κάτοψης του κτηρίου. Με βάση τα παραπάνω, όταν αναφερόμαστε σε πραγματικές συντεταγμένες θα εννοούμε τις συντεταγμένες σε μέτρα όπως προκύπτουν με βάση την ανωτέρω ανάλυση. Αντίθετα οι συντεταγμένες εικόνας αναφέρονται στα pixels της εικόνας. Επισημαίνεται ότι για να μην υπάρχουν παραμορφώσεις της εικόνας και προβλήματα απεικόνισης θα πρέπει να διατηρείται η αναλογία ανάμεσα στις δύο διαστάσεις μεταξύ των πραγματικών συντεταγμένων και των συντεταγμένων εικόνας. Έτσι αφού οι διαστάσεις του κτηρίου είναι 86m μήκος και 37m πλάτος (λόγος 86/37=2,32), επιλέγοντας 800 pixels για μήκος εικόνας προκύπτει 344 pixels για το πλάτος. Για αυτό και στο mapfile έχει οριστεί ως size της εικόνας το 800x344.

Προκειμένου να οριστεί το ανωτέρω σύστημα συντεταγμένων ακολουθήθηκε η εξής τεχνική:

Η κάτοψη του κτηρίου σαρώθηκε ώστε να δημιουργηθεί η αντίστοιχη εικόνα σε jpeg μορφή. Στη συνέχεια χρησιμοποιήθηκε το πρόγραμμα autocad προκειμένου, έχοντας ως βάση την jpeg εικόνα, να σχηματίσουμε τα βασικά σχήματα της κάτοψης σε ανυσματική μορφή (dxf αρχείο). Επισημαίνεται ότι κατά τη δημιουργία της ανυσματικής εικόνας λήφθηκαν υπόψη οι πραγματικές διαστάσεις των χώρων (αίθουσες, διάδρομοι κλπ) της κάτοψης. Στη συνέχεια η ανυσματική εικόνα μετατοπίστηκε σε σύστημα

συντεταγμένων με σημείο (0,0) την κάτω αριστερή γωνία της εικόνας. Έπειτα χρησιμοποιώντας το πρόγραμμα CAD2Shape στην έκδοση 3.0, το οποίο διατίθεται δωρεάν ως demo, μετατρέψαμε το dxf αρχείο σε shapefile. Ο μετέπειτα χειρισμός των shapefiles έγινε με το δωρεάν πρόγραμμα Quantum GIS. Επειδή οι συντεταγμένες και οι διαστάσεις είχαν επακριβώς οριστεί στο autocad με ακρίβεια τα αντίστοιχα shapefiles αναγνωρίστηκαν με το ίδιο σύστημα συντεταγμένων και τις ίδιες διαστάσεις. Μέσω του Quantum GIS υπάρχει η δυνατότητα ορισμού attributes στα shapefiles δημιουργίας επιπλέον layers και γενικότερα περαιτέρω ανάπτυξης ανυσματικών χαρτών μορφής shapefile. Έχοντας καλά ορισμένο από πλευράς διαστάσεων το βασικό σχήμα του ορόφου του πανεπιστημίου το οποίο ονομάσαμε Ground_shp και το ορίσαμε ως τύπου πολυγώνου, μπορέσαμε να δημιουργήσουμε επιπλέον layers, τύπου σημείου (με εξαίρεση το layer grCor το οποίο είναι τύπου γραμμής). Τα layers αυτά έχουν συγκεκριμένα attributes με πιο σημαντικά τα X, Y που παίρνουν ως τιμές τις πραγματικές συντεταγμένες του σημείου, το FLOOR που αναφέρεται στον όροφο του κτηρίου που βρίσκεται το συγκεκριμένο σημείο, το KIND που σχετίζεται με το είδος του σημείου (π.χ. Turn point, junction, end point) και το id του σημείου το οποίο είναι ένας αύξων αριθμός μοναδικός για κάθε σημείο. Το σύνολο των layers όπως τελικά ορίστηκαν φαίνονται στον Πίνακας 3 παρακάτω:

Πίνακας 3: Layers του χάρτη ισόγειας κάτοψης

Όνομα layer	Περιγραφή	Είδος	ID
Ground_shp	Βασικό σχήμα ορόφου	Polygon	-
grCor	Διάδρομοι	Line	-
grAll	Navigational Point	Point	1 - 33
grCD	Πόρτες Διαδρόμων	Point	34 – 36
grCOA	Είσοδοι σε κλειστές ή ανοικτές περιοχές	Point	37 – 43
grCR	Σημεία για ράμπες στους διαδρόμους	Point	44 - 47
grCS	Σκάλες σε διαδρόμους	Point	48 – 51
grEE	Είσοδοι – Έξοδοι κτηρίου	Point	52 – 54

grRo	Δωμάτια Ορόφου	Point	55 – 91
grEl	Ασανσέρ	Point	92
grS	Σκάλες που συνδέουν ορόφους	Point	93 – 96
grale	Βοηθητικά σημεία	Point	140 – 222
grObP	Εμπόδια	Point	223 - 227

Αναλυτικά τα attributes του κάθε layer μπορεί κάποιος να τα δει κάποιος είτε μέσω του Quantum GIS είτε ανοίγοντας με το Excel το αρχείο της μορφής «όνομα».dbf, όπου «όνομα» το όνομα του αντίστοιχου layer. Επισημαίνεται ότι τα αρχεία αυτά αποτελούν τμήμα των shapefiles τα οποία διαβάζει ο mapserver προκειμένου να σχηματίσει τους αντίστοιχους χάρτες. Το σύνολο των shapefiles παραδίδεται μαζί με την παρούσα διπλωματική σε ηλεκτρονική μορφή.

6.2.2. Ανάλυση κώδικα mapfile αρχείου

Το mapfile αρχείο που δημιουργήθηκε για τις ανάγκες της εφαρμογής είναι το «UniversityFinal.map». Πριν ξεκινήσουμε την ανάλυση του κώδικα του UniversityFinal.map κρίνεται σκόπιμο να αναφερθεί ότι οι λέξεις κλειδιά του mapfile, που έχουν αναπτυχθεί στο προηγούμενο κεφάλαιο, και οι τιμές τους δεν είναι case snensitive. Σ' αυτή την εργασία όμως επιλέξαμε για λόγους σαφήνειας να γράφουμε τις λέξεις κλειδιά με κεφαλαία και τις τιμές τους με μικρά. Πρέπει όμως να τονιστεί ότι η κατάσταση των γραμμάτων είναι σημαντική κατά την αλληλεπίδραση με χωρικές βάσεις δεδομένων.

Προχωρώντας με την ανάλυση του κώδικα, στη γραμμή 1 η λέξη κλειδί NAME ορίζει ότι η βάση για όλες τις εικόνες που θα δημιουργήσει ο Mapserver θα είναι UniversityFinal. Η λέξη κλειδί UNITS ορίζει τις μονάδες μέτρησης του χάρτη σε μέτρα. Η γεωγραφική έκταση (η ορθογώνια περιοχή που καλύπτεται από τον χάρτη) ορίζεται από την λέξη κλειδί EXTENT στη γραμμή 3. Η ορθογώνια περιοχή καθορίζεται από τις συντεταγμένες των σημείων των αντίθετων γωνιών (της κάτω αριστερής και της πάνω δεξιάς). Οι διαστάσεις του τελικού χάρτη θα είναι 800 X 344 pixels, όπως ορίζεται από την λέξη κλειδί SIZE. Στη γραμμή 5, το IMAGECOLOR καθορίζει ότι το χρώμα του φόντου για την map εικόνα θα είναι το λευκό και στη γραμμή 6 το IMAGETYPE καθορίζει ότι η μορφή της δημιουργούμενης εικόνας θα είναι SVG. Τέλος η θέση των shapefiles

καθορίζεται στη γραμμή από τη λέξη κλειδί SHAPEPATH στη γραμμή 7 και στη γραμμή 8 καθορίζεται η θέση του αρχείου fontset.txt, που κάνει την αντιστοίχιση της ονομασίας (alias) της γραμματοσειράς με τη θέση της.

```
NAME "UniversityFinal"

UNITS meters

EXTENT 0 0 86 37

SIZE 800 344

IMAGECOLOR 255 255 255

IMAGETYPE svg

SHAPEPATH "C:/Mapdata/finalUniversityData/Ground/"

FONTSET "C:/ms4w/Apache/htdocs/fontset.txt"
```

Στις γραμμές 10 έως και 16 ορίζεται και ονομάζεται η μορφή του παραγόμενου αρχείου με τη χρησιμοποίηση του OUTPUTFORMAT object, το οποίο επεκτείνει την έννοια της μορφής της εικόνας svg για να περιλάβει λεπτομέρειες για τη δομή της εικόνας και ακόμα ποιος driver γραφικών πρέπει να χρησιμοποιηθεί για να παραχθεί μια εικόνα.

```
OUTPUTFORMAT
    NAME svg
    MIMETYPE "image/svg+xml"
    DRIVER "svg"
    FORMATOPTION "COMPRESSED_OUTPUT=FALSE"
    FORMATOPTION "FULL_RESOLUTION=FALSE"
END
```

Εκτός από τον ορισμό του OUTPUTFORMAT object για να παραχθεί μια svg εικόνα πρέπει να δοθούν έγκυρες τιμές στις λέξεις κλειδιά IMAGEPATH και IMAGEURL στο WEB object του mapfile. Το WEB object στο συγκεκριμένο mapfile βρίσκεται στις γραμμές 18 έως και 21. Η λέξη κλειδί IMAGEPATH υποδεικνύει στον Mapserver που να τοποθετήσει τις εικόνες χαρτών που παράγει και η λέξη κλειδί IMAGEURL καθορίζει ένα URL που λέει στον browser που να κοιτάξει για να ανακτήσει την εικόνα. Ο Mapserver θα συμπεριλάβει το URL στην σελίδα πριν σταλεί πίσω στον browser. Σ' αυτό το σημείο πρέπει να τονιστεί ότι το IMAGEPATH string είναι ένα απόλυτο μονοπάτι στον local host

καθώς το IMAGEURLκαθορίζει τη θέση λαμβάνοντας υπόψη το DocumentRoot του web server.

```
WEB
    IMAGEPATH "/ms4w/tmp/ms_tmp/"
    IMAGEURL  "/ms_tmp/"
END
```

Με βάση τις παραπάνω πληροφορίες ο Mapserver ξέρει τι είδος εικόνας να παράγει, τι μέγεθος και χρώμα φόντου να δώσει σε αυτή και πώς να απεικονίσει το χάρτη που δημιουργεί σε μια ιστοσελίδα. Αυτό που δεν γνωρίζει ακόμη είναι τι να σχεδιάσει και πώς να το σχεδιάσει. Υπεύθυνο για αυτές τις εργασίες του Mapserver είναι το LAYER object, το οποίο εισάγει τα layers του χάρτη. Ένα layer, όπως έχει αναφερθεί και παραπάνω, αναφέρεται σε μια μοναδική ομάδα δεδομένων και περιέχει μια ομάδα στοιχείων που θα αποδοθούν μαζί σε μια συγκεκριμένη κλίμακα χρησιμοποιώντας μια συγκεκριμένη προβολή Στη συνέχεια του κώδικα αναπτύσσονται τα layers του χάρτη.

Στις γραμμές 26 έως και 53 ορίζεται το πρώτο layer του χάρτη, που αποτελεί το σχήμα του 1^{ου} ορόφου. Η λέξη κλειδί NAME στη γραμμή 27 καθορίζει ένα όνομα για το layer. Στην επόμενη γραμμή, η λέξη κλειδί DATA προσδιορίζει το όνομα του shapefile που θα αποδοθεί σ' αυτό το layer. Η τιμή που συνδέεται με αυτή τη λέξη κλειδί είναι στην πραγματικότητα το σχετικό μονοπάτι από το SHAPEPATH. Στη συνέχεια του κώδικα, στη γραμμή 29, εμφανίζεται η λέξη κλειδί STATUS, η οποία αποφασίζει αν ένα layer θα αποδοθεί ή όχι και αν η κατάσταση του μπορεί να αλλάξει. Στην προκειμένη περίπτωση το STATUS είναι on, που σημαίνει ότι το layer αρχικά θα απεικονίζεται αλλά σε μεταγενέστερη κατάσταση να μην εμφανίζεται αν αυτό επιλέξει ο χρήστης. Η λέξη κλειδί TYPE καθορίζει ότι ο τύπος του layer "Ground_shp" θα είναι polygon.

```
LAYER
  NAME "Ground_shp"
  DATA "Ground_shp"
  STATUS on
  TYPE polygon
```

Στις γραμμές 31 έως 33 εμφανίζονται οι λέξεις κλειδιά LABELCACHE, LABELITEM και CLASSITEM. Η λέξη κλειδί LABELCACHE δίνει την δυνατότητα στον Mapserver να αποθηκεύει τις ετικέτες των layers, των οποίων το LABELCACHE είναι on, και να τις

παραδίδει ταυτόχρονα αφού αυτά έχουν σχηματιστεί. Εκτός από το χρόνο που θα παραδοθεί μια ετικέτα πρέπει και να ονομαστεί. Αυτό ακριβώς κάνει η λέξη κλειδί LABELITEM, δίνει όνομα στην κάθε ετικέτα του layer. Το LABELITEM έχει την τιμή “id” πράγμα που σημαίνει ότι η ετικέτα κάθε πολυγώνου θα πάρει το όνομα της από το attribute “id” του shapefile (layer). Η λέξη κλειδί CLASSITEM αναγνωρίζει το όνομα του attribute που θα χρησιμοποιηθεί για να ταξινομήσει τα features.

```
LABELCACHE on
    LABELITEM "id"
    CLASSITEM "id"
```

Στη συνέχεια του κώδικα ορίζεται το class object του layer, στο οποίο περιέχονται οι οδηγίες, που καθοδηγούν τον Mapserver, ως προς τον τρόπο παράδοσης των features σ’ ένα layer. Στο layer “Ground_shp” υπάρχει μία κλάση, όμως ο αριθμός των κλάσεων μπορεί να είναι και μεγαλύτερος σ’ ένα layer. Αυτό συμβαίνει όταν για παράδειγμα σ’ ένα layer που αναπαριστά δρόμους θέλουμε να ξεχωρίσουμε τους αυτοκινητόδρομους από τους μικρούς επαρχιακούς δρόμους, οπότε και θα τους κάνουμε μεγαλύτερους και με διαφορετικό χρώμα. Η μοναδική λοιπόν κλάση του layer “Ground_shp” προκαλεί τον Mapserver να επιλέξει όλα τα feature για να τα αποδώσει στη συνέχεια.

Η λέξη κλειδί NAME της κλάσης είναι η ετικέτα που θα εμφανιστεί στη λεζάντα που σχετίζεται με τον χάρτη. Το style object, που εμφανίζεται στις γραμμές 36 έως 40, περιέχει τις παραμέτρους που περιγράφουν τον τρόπο με τον οποίο θα σχεδιασθούν τα πολύγωνα της κλάσης. Με τη λέξη κλειδί SIZE καθορίζεται το ύψος του πολυγώνου, με την COLOR καθορίζεται το χρώμα το οποίο θα γεμίσει το πολύγωνο και η OUTLINECOLOR καθορίζει το χρώμα του περιγράμματος των πολυγώνων.

```
CLASS
        NAME "id"
        STYLE
            SIZE 1
            COLOR 211 255 190
            OUTLINECOLOR 0 0 0
        END
```

Επίσης μέσα στο class object υπάρχει και το label object, το οποίο παραδίδεται με την κλάση και καθορίζει τον τύπο της γραμματοσειράς, το μέγεθος , την θέση και το χρώμα της ετικέτας. Το TYPE, στη γραμμή 42, καθορίζει τον τύπο της γραμματοσειράς που χρησιμοποιείται για να αποδώσει την ετικέτα. Οι επιλογές που υπάρχουν είναι δύο: οι bitmapped και οι TrueType γραμματοσειρές. Οι bitmapped δημιουργούνται εσωτερικά και είναι οι default γραμματοσειρές του mapserver και οι TrueType πρέπει να εγκατασταθούν. Στον κώδικα μας ο τύπος της γραμματοσειράς που έχει επιλεγεί είναι TrueType, οπότε είναι απαραίτητο να καθοριστεί και η γραμματοσειρά που θα χρησιμοποιηθεί. Στην επόμενη γραμμή η λέξη κλειδί FONT παίρνει την τιμή “arial”, που σημαίνει ότι ο Mapserver ξέρει ότι η γραμματοσειρά που εμφανίζεται με το string “arial” στο FONTSET αρχείο θα χρησιμοποιηθεί γι’ αυτή την ετικέτα. Με το SIZE ορίζεται το μέγεθος της ετικέτας σε 8 pixels, με το COLOR ορίζεται το χρώμα σε μαύρο και με το OUTLINECOLOR το χρώμα περιγράμματος σε λευκό. Στις επόμενες γραμμές η λέξη κλειδί MINDISTANCE ορίζει τον αριθμό τον pixels μεταξύ ίδιων ετικετών σε 5, η λέξη κλειδί POSITION ορίζει τη θέση της ετικέτας στο κέντρο του καθορισμένου τετραγώνου γι’ αυτή (cc – center center) και το MINFEATURESIZE ορίζει ότι το μέγεθος (σε pixels) της μικρότερης ετικέτας που θα σχεδιασθεί θα είναι 5. Στη γραμμή 50 η λέξη κλειδί ορίζει σαν χαρακτήρα αλλαγής γραμμής μιας ετικέτας το space (“ “).

```
LABEL
        TYPE truetype
        FONT "arial"
        SIZE 8
        OUTLINECOLOR 255 255 255
        COLOR 0 0 0
        MINDISTANCE 5
        POSITION cc
        MINFEATURESIZE 5
        WRAP ' '
END
```

Στη συνέχεια του κώδικα περιγράφονται με τον ίδιο τρόπο τα υπόλοιπα layers που αναφέρονται στον Πίνακα 3. Δηλαδή ξεκινάει πάλι ένα layer object για κάθε ένα από τα υπολειπόμενα layers με τη λέξη κλειδί LAYER και κλείνει με τη λέξη κλειδί END. Πάλι

μέσα στο layer object περιέχεται ένα class object, το οποίο περιέχει ένα label object για το σχεδιασμό των ετικετών.

6.2.3. Ανάλυση κώδικα php αρχείου

Ο κώδικας σε php/mapscript που τρέχει στον server περιέχεται στο αρχείο «getUniversityMap.php». Η γραμμή 1 ξεκινά το php script. Στη γραμμή 2 καθορίζεται ότι ο server θα επιστρέψει στον client ένα αντικείμενο το οποίο θα είναι τύπου svg. Στις επόμενες γραμμές καθορίζεται η διαδρομή και το όνομα του mapfile το οποίο θα πρέπει να διαβάσει η mapscript για τη δημιουργία του χάρτη.

```
<?php

header("Content-type: image/svg+xml");

$map_path = "C:/ms4w/Apache/htdocs/UniversityFinal/";

$map_file = "UniversityFinal.map";

$img_path = "/ms4w/tmp/ms_tmp/";
```

Στις γραμμές 15 έως και 21 αποθηκεύονται σε αντίστοιχες μεταβλητές οι παράμετροι κλήσης του server.

```
//get URL parameters

$layername = $_GET['layername'];

$xmin = intval($_GET['xmin']);

$xmax = intval($_GET['xmax']);

$ymin = intval($_GET['ymin']);

$ymax = intval($_GET['ymax']);

$timestamp = $_GET['timestamp'];
```

Ουσιαστικά είναι οι παράμετροι που έχουν αποσταλεί από τον client και αφορούν το layer που έχει ζητηθεί να απεικονισθεί, το παράθυρο θέασης του layer και μια μεταβλητή σχετική με το χρόνο που έγινε η κλήση από τον client. Το παράθυρο θέασης αναφέρεται στις γεωγραφικές συντεταγμένες (x,y), που στην περίπτωσή μας εκφράζονται σε μέτρα, της κάτω αριστερής και της πάνω δεξιάς γωνίας του παραλληλογράμου τμήματος που έχει ζητηθεί να προβληθεί από τον client (περιπτώσεις zoom in/out και pan). Οι αποστάσεις μεταξύ των συντεταγμένων του

παραθύρου θέασης αποτελούν και το extent του χάρτη που θα αποστείλει ο server. Το μέγιστο extent είναι 86m μήκος και 37m πλάτος και είναι οι εξωτερικές διαστάσεις του χώρου που απεικονίζουμε. Το σημείο (0,0) θεωρείται η κάτω αριστερή γωνία του χάρτη όταν αυτός είναι στο μέγιστο extent (πλήρης χάρτης χωρίς zoom).

Στις γραμμές 33-52 ορίζεται η συνάρτηση img2map(), ενώ στις γραμμές 56-69 η συνάρτηση map2img(). Οι συναρτήσεις αυτές χρησιμοποιούνται όταν απαιτείται μετατροπή των συντεταγμένων εικόνας (pixels στην οθόνη) σε πραγματικές συντεταγμένες και αντίστροφα. Η mapscript συνάρτηση zoomrectangle() απαιτεί ως παράμετρο να δοθεί το παράθυρο που θα γίνει zoom, σε συντεταγμένες εικόνας. Από την άλλη ο client αποστέλλει το παράθυρο θέασης (στο οποίο θέλει να γίνει zoom) σε πραγματικές συντεταγμένες. Έτσι θα πρέπει να υπάρχουν οι αντίστοιχες συναρτήσεις μετατροπής από το ένα σύστημα στο άλλο.

```
//conversion of geographical coordinates (meters) to image coordinates (pixels)
function map2img($width,$height,$point,$ext) {
  // valid point required
  if ($point->x && $point->y){
    // find pixels per degree
    $ppd_x = $width/($ext->maxx - $ext->minx);
    $ppd_y = $height/($ext->maxy - $ext->miny);
    // calculate image coordinates
    $p[0] =  $ppd_x * ($point->x - $ext->minx);
    $p[1] =  $ppd_y * ($point->y - $ext->miny);
    settype($p[0],"integer");
    settype($p[1],"integer");
  }
  return $p;
} // end map2img
```

Η μέθοδος που χρησιμοποιείται είναι απλή. Η map2img λαμβάνει ως παραμέτρους το μήκος και πλάτος του χάρτη σε pixels, τις συντεταγμένες του σημείου σε μέτρα που

θέλουμε να μετατρέψουμε από μέτρα σε pixels και το extent του χάρτη σε μέτρα. Υπολογίζει τα pixel ανά μέτρο του χάρτη και στη συνέχεια πολλαπλασιάζοντας τον αριθμό αυτό με την απόσταση του σημείου που θέλουμε να μετατρέψουμε από την κάτω αριστερή γωνία του παραθύρου θέασης βρίσκουμε τα αντίστοιχα pixels. Στις γραμμές 63-64 αποθηκεύονται οι τιμές των pixels ανά διάσταση σε έναν πίνακα. Στη συνέχεια οι τιμές αυτές ορίζονται ως τύπου integer, ώστε να απαλειφθούν και τα τυχόν δεκαδικά ψηφία και στη γραμμή 66 ο πίνακας που περιέχει τις ζητούμενες τιμές επιστρέφεται στη συνάρτηση που κάλεσε την map2img. Αντίστοιχα λειτουργεί και η συνάρτηση img2map.

Στη γραμμή 72 ορίζεται ένα νέο Map object βάσει του αρχείου mapfile που περιγράφεται παραπάνω. Το extent του αντικειμένου αυτού είναι αυτό που ορίζεται στο mapfile ενώ τα layers που θα εμφανίζει είναι αυτά για τα οποία το STATUS στο mapfile έχει οριστεί ως Default ή On.

```
// Retrieve mapfile and create a map from it
$map = ms_newMapObj($map_path.$map_file);
```

Στις γραμμές 74-84 επιλέγονται τα layers που θα απεικονιστούν από τον mapserver με βάση το τι ζητήθηκε από τον client. Ουσιαστικά εξετάζει όλα τα layers και εφόσον το όνομα κάποιου από αυτά είναι ίδιο με τη μεταβλητή στην οποία είχε αποθηκευτεί το layer που είχε ζητήσει ο client ($layername) το θέτει ON διαφορετικά το layer τίθεται OFF.

```
//set the map layers to ON (will be drawn) or OFF (not drawn) depending
//on the chosen layername variable
for ($layerindex = 0; $layerindex < $map->numlayers; $layerindex++) {
 $lay = $map->getLayer($layerindex);
 if ($layername==$lay->name ) {
  $lay->set(status,MS_ON);
 } else {
  $lay->set(status,MS_OFF);
 }
}
```

Στις γραμές 86-106 τίθεται το extent σε μέτρα του χάρτη με βάση το παράθυρο θέασης που δόθηκε ως παράμετρος από τον client και γίνεται zoom στο παράθυρο αυτό. Για το σκοπό αυτό χρησιμοποιούνται οι mapscript συναρτήσεις setExtent και zoomrectangle με τις αντίστοιχες απαιτούμενες μεταβλητές.

```
// Set the map to the extent retrieved from the form
$map->setextent($extent[0],$extent[1],$extent[2],$extent[3]);
// Save this extent as a rectObj, we need it to zoom.
$old_extent->setExtent($extent[0],$extent[1],$extent[2],$extent[3]);
$my_extent=ms_newRectObj();
$my_extent->setExtent($extent[0],$extent[1],$extent[2],$extent[3]);
$LLpoint=ms_newPointObj();
$URpoint=ms_newPointObj();
$LLpoint->setXY($extent[0]+0.01,$extent[1]+0.01);
$URpoint->setXY($extent[2],$extent[3]);
list($a,$b) = map2img($map->width,$map->height,$LLpoint,$old_extent);
list($c,$d) = map2img($map->width,$map->height,$URpoint,$old_extent);
$my_extent->setExtent($a,$b,$c,$d);
$map->zoomrectangle($my_extent,$map->width,$map-
>height,$old_extent,$max_extent);
```

Επισημαίνεται ότι στη γραμμή 98 ορίζουμε το σημείο της κάτω αριστερής γωνίας. Παρατηρούμε ότι στο σημείο αυτό δίνουμε την τιμή ($extent[0]+0.01, $extent[1]+0.01). Υπενθυμίζουμε ότι η τιμή ($extent[0], $extent[1]) στην περίπτωση που δεν έχει γίνει zoom είναι (0,0). Επειδή παρατηρήθηκε ότι μέσω της mapscript δεν ήταν δυνατό να γίνει zoom εάν δινόταν αυτή η τιμή κρίθηκε αναγκαίο να προσθέτουμε μια μικρή μη μηδενική τιμή. Αυτό δεν αλλοιώνει το αποτέλεσμα, αφού στη συνέχεια με τη συνάρτηση map2img γίνεται στρογγυλοποίηση σε integer. Στην περίπτωση που ο χάρτης μας δεν ξεκινά από το (0,0) αλλά από μια μη μηδενική τιμή δεν απαιτείται να προσθέσουμε το 0.01. Στην τελική υλοποίηση της εφαρμογής τα shapefiles έγιναν με τέτοιο τρόπο ώστε

η αρχή των αξόνων να μην είναι το (0,0) αλλά το (0.01,0.01), οπότε επί της ουσίας δεν χρειάζεται να προσθέσουμε κάποιον αριθμό.

Στις γραμμές 108-113 διαβάζεται το αρχείο path.txt στο οποίο έχουν καταγραφεί τα id των σημείων τα οποία περιλαμβάνει η διαδρομή που θέλει ο χρήστης.

```
//save to a variable the text file that contains the required navigational path
//the path is given by point ids separated commas
$path_file = "path.txt";
//read the path file and store it  to an array named $path
$path_matrix=file_get_contents($path_file);
$path=explode(",",$path_matrix);
```

Ουσιαστικά τα ids αποθηκεύονται στον πίνακα $path προκειμένου να χρησιμοποιηθούν στη συνέχεια για τη δυναμική δημιουργία ενός layer που θα περιλαμβάνει ευθύγραμμα τμήματα που θα ενώνουν τα σημεία με τα συγκεκριμένα ids. Κάθε φορά που αλλάζουμε το αρχείο path.txt, αλλάζει και το layer της διαδρομής (route). Σε επόμενη φάση υπάρχει η δυνατότητα χρησιμοποίησης κάποιου αλγόριθμου εύρεσης βέλτιστης διαδρομής μεταξύ των ζητούμενων σημείων το αποτέλεσμα του οποίου θα καταγράφεται στο path.txt ώστε να απεικονιστεί στο χάρτη.

Στις γραμμές 122-140 επεξεργαζόμαστε το layer το οποίο έχει ζητήσει για απεικόνιση ο client. Αυτό που πραγματοποιείται είναι η εύρεση όλων των σχημάτων (shapes) που περιέχονται στο παράθυρο θέασης του συγκεκριμένου layer προκειμένου να αποθηκευτούν στους πίνακες $kind, $pointsid και $ids τα attributes KIND, FLOOR και ID των σχημάτων αυτών. Ο λόγος που αποθηκεύουμε τις τιμές αυτές είναι προκειμένου να χρησιμοποιηθούν από τις συναρτήσεις insertIDs και insertGroundIDs όπως θα εξηγηθεί παρακάτω. Δεδομένου ότι τα layers grCor και Route δεν περιέχουν τα ανωτέρω attributes γίνεται ο έλεγχος της γραμμής 122 (if ($layername !="grCor" && $layername !="Route")*).*

```
//If $layername is a point layer then we should get the id of each point on that
//layer, store them in the array $ids[] in order to
// insert these values later in the svg file, since mapserver doesn't insert these ids in the svg by itself.
//*****************************************************************************
```

```
if ($layername !="grCor" && $layername !="Route")
{
 $i=0;
 $this_layer= $map->getLayerByName($layername);
 $status=$this_layer->open();
 $status=$this_layer->whichShapes($map->extent);
 if ($status == MS_SUCCESS)
 {
 //every shape is a point object. We read its ID feature and save it in an array
  while ( $shape=$this_layer->nextShape())
  {
   $kind[$i]=$shape->values["KIND"];
   $pointsid[$i]=$shape->values["FLOOR"];
   $ids[$i]=$shape->values["ID"];
   $i++;
  }
 }
 $this_layer->close;
}
```

Ιδιαίτερη μνεία θα πρέπει να γίνει στον έλεγχο if ($status==MS_SUCCESS) ο οποίος δε φαίνεται να χρησιμοποιείται στη βιβλιογραφία του mapserver οπότε αρχικά δεν είχε ενσωματωθεί στον κώδικά μας με αποτέλεσμα να προκαλούνται προβλήματα στη λειτουργία της εφαρμογής. Βάσει του ορισμού των συναρτήσεων του mapserver η αμέσως προηγούμενη του ελέγχου αυτού κλήση συνάρτησης (δηλαδή η $this_layer->whichShapes($map->extent)) είναι απαραίτητη προκειμένου να δουλέψει το while loop. Η συνάρτηση $this_layer->whichShapes($map->extent); Επιστρέφει MS_SUCCESS ή MS_FALSE. Ακόμα και εάν εκτελέσουμε το wile loop χωρίς να κάνουμε έλεγχο της $status η εφαρμογή δουλεύει κανονικά. Εκτελώντας την εφαρμογή υπό διάφορες συνθήκες (συνδυασμοί επιλογών layers και zoom) παρατηρήθηκε ότι κάποιες φορές η εφαρμογή εμφάνιζε μηνύματα σφάλματος που είχαν να κάνουν με

αδυναμία σύνδεσης με τον server. Λόγω χρήσης της τεχνολογίας AJAX το πρόγραμμα συνέχιζε να λειτουργεί οπότε παρατηρήσαμε ότι αλλάζοντας το συνδυασμό zoom και επιλογής layer συνδεόταν κανονικά στο server και έδινε τα ζητούμενα αποτελέσματα. Τελικά αποδείχθηκε ότι η αδυναμία σύνδεσης με το server οφειλόταν στο ότι έληγε το χρονικό διάστημα αναμονής απάντησης από τον server. Αυτό συνέβαινε στην περίπτωση που είχε επιλεγεί να εμφανιστεί κάποιο layer το οποίο όμως στο συγκεκριμένο παράθυρο θέασης που είχαμε επιλέξει να κάνουμε zoom δεν διέθετε κανένα σχήμα. Το αποτέλεσμα ήταν ο βρόχος while να ψάχνει για σχήματα εκεί όπου δεν υπήρχαν με αποτέλεσμα ο server να μη δίνει καμία απάντηση στον client. Για το λόγο αυτό είναι απαραίτητος ο έλεγχος της μεταβλητής $status έτσι ώστε εφόσον το παράθυρο θέασης δεν περιλαμβάνει shapes του συγκεκριμένου layer ο while βρόχος να μην ξεκινά.

Στις γραμμές 145 – 174 ορίζονται οι συναρτήσεις insertIDs και insertGroundIDs με τις οποίες εισάγονται τα ids στα σχήματα των layer του svg αρχείου. Σαν παραμέτρους δέχονται τους πίνακες $kind, $pointsid και $ids που δημιουργήσαμε παραπάνω καθώς και το svg αρχείο που δημιουργεί ο mapserver. Η λειτουργία των δύο συναρτήσεων είναι παρόμοια οπότε παρακάτω παραθέτουμε μόνο τη μία από αυτές.

```
//inserts the ids in the svg point layers
function insertIDs($Kindarray,$PointIDArray,$IDsArray,$pointslayerFile)
{
 $data = file($pointslayerFile);
 // find the size of rows of the svg file
 $count = count($data);
 //insert ids from ids array where the point ads where stored, to svg file
 for ($i = 0; $i < $count; $i++)
 {
  //insertion is made by replacement of ellipse with ellipse+id
  $data[$i+4]=str_replace("ellipse", "ellipse id=\"".$Kindarray[$i]."_". $PointIDArray[$i].
"_".$IDsArray[$i]."\"", $data[$i+4]);
 }
 //save the new svg with the ids
```

```
 file_put_contents($pointslayerFile, $data) or die('Could not write to file');
}
```

Ο λόγος που χρησιμοποιούνται οι συναρτήσεις αυτές είναι ο εξής: Ο mapserver όπως θα δούμε παρακάτω για κάθε layer που του ζητείται φτιάχνει ένα svg αρχείο το οποίο αποστέλλει στον client. Η μορφή του svg είναι η εξής (δίνεται το layer grAll):

```
<?xml version="1.0" encoding="UTF-8"?>
<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1" width="800" height="344" id="grAll" timestamp="1205347040480">

<!-- START LAYER grAll -->
<ellipse cx="515" cy="340" rx="2" ry="2" fill="#0000ff"/>
<ellipse cx="475" cy="341" rx="2" ry="2" fill="#0000ff"/>
<ellipse cx="569" cy="235" rx="2" ry="2" fill="#0000ff"/>
<ellipse cx="590" cy="235" rx="2" ry="2" fill="#0000ff"/>
….
</svg>
```

Παρατηρούμε ότι από τη γραμμή *<!-- START LAYER grAll -->* και κάτω κάθε σχήμα του layer (στην περίπτωσή μας είναι ένα point) αναπαρίσταται σε μια γραμμή αλλά δεν δίνεται κανένα id για το σχήμα αυτό. Έτσι όμως ο client δεν έχει τη δυνατότητα να το χειριστεί περαιτέρω εφόσον το svg δεν περιέχει τα ids των σχημάτων. Δεδομένου ότι δεν είχαμε τη δυνατότητα να επέμβουμε στις συναρτήσεις του mapserver που καθορίζουν τον τρόπο δημιουργίας του svg από τα shapefiles και παρατηρώντας ότι η σειρά των παραπάνω γραμμών που αναπαριστούν τα σημεία είναι ίδια με τη σειρά των ids των shapefiles που διαβάζει ο mapserver, δημιουργήσαμε τις συναρτήσεις insertIds με τις οποίες διαβάζεται το svg που έχει δημιουργηθεί και σε κάθε γραμμή εισάγουμε το id το οποίο προκύπτει ως ένωση (concatenation) των πινάκων $kind, $pointsid και $ids που περιέχουν τα αντίστοιχα attributes των shapefiles. Έτσι το αποτέλεσμα είναι το νέο svg αρχείο να είναι της μορφής:

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

<svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1" width="800" height="344" id="grAll" timestamp="1205347040480">

<!-- START LAYER grAll -->

<ellipse id="Junction_0_1" cx="515" cy="340" rx="2" ry="2" fill="#0000ff"/>

<ellipse id="Junction_0_2" cx="475" cy="341" rx="2" ry="2" fill="#0000ff"/>

<ellipse id="Junction_0_3" cx="569" cy="235" rx="2" ry="2" fill="#0000ff"/>

<ellipse id="Turn_Point_0_4" cx="590" cy="235" rx="2" ry="2" fill="#0000ff"/>

…

</svg>

Στις γραμμές 176-204 ελέγχεται εάν απαιτείται η απεικόνιση του layer της διαδρομής (route) και εφόσον αυτό ισχύει γίνεται εύρεση των πραγματικών συντεταγμένων των σχημάτων (points) των οποίων τα ID έχουν διαβαστεί από το αρχείο path.txt που αναφέρθηκε παραπάνω. Επισημαίνεται ότι τα IDs αυτά έχουν ήδη διαβαστεί και αποθηκευτεί στον πίνακα $path. Επίσης θα πρέπει να τονιστεί ότι η διαδικασία αυτή είναι ανεξάρτητη από την προηγούμενη με την οποία κάναμε εισαγωγή των IDs των σχημάτων στο svg αρχείο. Αυτό σημαίνει ότι ακόμα και εάν η προηγούμενη διαδικασία δεν γινόταν η εύρεση των πραγματικών συντεταγμένων με βάση δοθέντα IDs παραμένει εφικτή. Για να γίνει αυτό κατανοητό σημειώνεται ότι η συγκεκριμένη μετατροπή γίνεται στο map object και όχι στο svg αρχείο. Ο mapserver γνωρίζει και μέσω της mapscript μπορεί να χειριστεί τα ids του map object, απλά δεν τα περνάει στο svg όταν το δημιουργεί.

```
//If the user chose to see the route
if ($layername == 'Route')
{
 $i=0;
//Read all the point layers
 for ($layerindex = 0; $layerindex < $map->numlayers; $layerindex++)
 {
```

```
  $lay = $map->getLayer($layerindex);
  if ($lay->name != "Ground_shp" && $lay->name !="grCor" && $lay->name !="Route")
  {
   $status=$lay->open();
//for each shape (point) get its id and if this matches with the id read by the path array in which the desired route is stored
//get the geographic X,Y coordinates (meters)  of that point. These coordinates are stored to the XYarray and they will be used
//to draw the path lines and the points of the route
    for ($i=0;$i<count($path);$i++)
    {
     $status=$lay->whichShapes($my_extent);
     while ($shape=$lay->nextShape())
     {
      $element=$shape->values["KIND"]."_".$shape->values["POINTID"]."_".$shape->values["ID"];
       if ( $path[$i] == $element)
       {
        $XYarray[$i]=((double)$shape->values["X"]).",".((double)$shape->values["Y"]).",".($shape->values["ID"])."\n";
       }
     }
    }
    $lay->close;
  }
 }
```

Η παραπάνω διαδικασία είναι απαραίτητη για την απεικόνιση της διαδρομής. Είπαμε ότι η διαδρομή ουσιαστικά δημιουργείται δυναμικά ως ένα επιπλέον layer. Με χρήση mapscript μπορούμε να φτιάξουμε το layer αυτό το οποίο θα είναι τύπου line. Η δημιουργία ενός ευθύγραμμου τμήματος προϋποθέτει την ύπαρξη δύο σημείων. Τα σημεία αυτά θα πρέπει να δοθούν με τη μορφή πραγματικών συντεταγμένων. Όπως είπαμε παραπάνω ανάμεσα στα attributes των layers των shapefiles είναι και οι πραγματικές τους συντεταγμένες (x,y). Με τον παραπάνω κώδικα αναζητούμε τα δοθέντα IDs (του path.txt) στα points των layers και εφόσον τα βρούμε αποθηκεύουμε τις πραγματικές συντεταγμένες των σημείων αυτών στον πίνακα XY.

Στις γραμμές 205-271 δημιουργείται ένα νέο layer το οποίο περιλαμβάνει τη γραμμή που ενώνει τα σημεία με πραγματικές συντεταγμένες αυτές του πίνακα XY. Επιπρόσθετα, επειδή η γραμμή αυτή μπορεί να τύχει να περνάει και πάνω από άλλα σημεία τα οποία όμως δεν είχαμε επιλέξει να είναι στη διαδρομή, για να αποφευχθεί τυχόν παρανόηση, τα σημεία που ανήκουν στη ζητούμενη διαδρομή χρωματίζονται εντονότερα στο χρώμα της υπόψη γραμμής (κόκκινο). Για να γίνει αυτό απλά στο layer της διαδρομής δημιουργούμε δυναμικά και σημεία (points) με πραγματικές συντεταγμένες που λαμβάνονται από τον πίνακα XY. Δηλαδή το layer της διαδρομής περιλαμβάνει τόσο τη γραμμή της διαδρομής όσο και τα σημεία αυτής.

Μετά τα παραπάνω έχουμε δημιουργήσει το τελικό map object που ο mapserver πρέπει να κάνει ανυσματική εικόνα τύπου svg και να στείλει στον client.

Στις γραμμές 276-282 δημιουργείται και αποθηκεύεται το ζητούμενο svg αρχείο.

```
// Drawing the map

// create unique names for map and reference images

$image_name = "University_".$layername.".svg";

//$image_url="/ms_tmp/".$image_name;

$imagefilename=$img_path.$image_name;


// Draw map image

$image=$map->draw();

$image->saveImage($imagefilename);
```

Εδώ θα πρέπει να σχολιαστεί ένα πρόβλημα που παρατηρήθηκε στη λειτουργία του mapserver κατά τη δημιουργία svg αρχείων. Με τη συνάρτηση *$image=$map->draw();* ο mapserver δημιουργεί το χάρτη με βάση το map object ενώ με την *saveImage($imagefilename);* τον αποθηκεύει. Στις περιπτώσεις των εικόνων raster η συνάρτηση $map->draw δημιουργεί τη raster εικόνα και η saveImage την αποθηκεύει. Στην περίπτωση όμως της svg εικόνας παρατηρούμε ότι η $map->draw όχι μόνο δημιουργεί την svg εικόνα αλλά συγχρόνως την αποθηκεύει με ένα όνομα που επιλέγει τυχαία. Στη συνέχεια με την *saveImage* η svg εικόνα αποθηκεύεται με το όνομα που έχουμε επιλέξει. Στην περίπτωση που δεν χρησιμοποιήσουμε την draw δεν δουλεύει η saveImage ενώ αν δεν χρησιμοποιήσουμε την saveImage δεν μπορούμε να εκτελέσουμε συγκεκριμένες λειτουργίες (όπως εισαγωγή ids) αφού δεν γνωρίζουμε το όνομα της svg εικόνας ώστε να το διαβάσουμε και να το επεξεργαστούμε. Το αποτέλεσμα είναι για κάθε layer να αποθηκεύονται δύο svg αρχεία όπως για παράδειγμα τα παρακάτω που αφορούν το layer Ground_shp:

47dc408d_d94_0.svg και University_Ground_shp.svg

Σημειώνεται επίσης ότι στον client αποστέλλεται το ένα από τα 2 svg αρχεία (αυτό του οποίου το όνομα μας είναι γνωστό).

Στις γραμμές 284-291 καλούμε τις αντίστοιχες συναρτήσεις για εισαγωγή των ids.

```
if ($layername != "Ground_shp" && $layername != "Route")
{
  //we insert  the id of the point object in the svg file
  insertIDs($kind,$pointsid,$ids,$imagefilename);
}elseif ($layername == "Ground_shp")
{
  insertGroundIDs($ids,$imagefilename);
}
```

Τέλος στις γραμμές 293 έως 318 αποστέλλουμε το svg αρχείο στον client αφού πρώτα του ενσωματώσουμε κάποια απαραίτητα στοιχεία.

```
//creation of a new xml document
```

```
$doc = new DomDocument();

//loading our svg image
$doc->Load($img_path.$image_name);

//getting the root element of the svg image which is an svg element
$root = $doc->documentElement;

//creation of id attribute with value of the $layername variable
$id_var = new DOMAttr("id", $layername);
//insertion of the id attribute to the root element
$root->appendchild($id_var);

//creation of timestamp attribute with value of the $timestamp variable
$timestamp_var = new DOMAttr("timestamp", $timestamp);

//insertion of the timestamp attribute to the root element
$root->appendchild($timestamp_var);

//creation of the XML output
$output=$doc->saveXML();
$doc->save($img_path.$image_name);

//sending the SVG/XML file to the user
echo $output;
```

Ήδη από την αρχή του php κώδικα δηλώθηκε στο header ότι ο server θα επιστρέψει ένα αρχείο τύπου svg (δηλαδή xml). Με κάθε κλήση στον server δημιουργείται ένα svg αρχείο που περιλαμβάνει ένα layer χωρίς όμως στο svg να υπάρχει ως attribute το id του layer αυτού. Επιπρόσθετα για την σωστή λειτουργία του client θα πρέπει στο svg αρχείο να εισάγουμε ως attribute και τη μεταβλητή $timestamp. Προκειμένου να υλοποιηθούν τα ανωτέρω κάνουμε χρήση των δυνατοτήτων που μας παρέχει η php να χειριστούμε ένα xml Dom Document. Ουσιαστικά δημιουργούμε ένα νέο DomDocument στο οποίο φορτώνουμε το svg αρχείο (εφόσον το svg είναι xml) και σε αυτό εισάγουμε τα επιθυμητά attributes.

Στο τέλος με τη συνάρτηση *echo $output;* το τελικό αποτέλεσμα αποστέλλεται στον client.

6.3. Τμήμα εφαρμογής στον client

Για την υλοποίηση του τμήματος της εφαρμογής στον client στηριχθήκαμε σε βιβλιοθήκες javascript και βασικών προγραμμάτων και τεχνικών υλοποίησης svg που περιγράφονται στην ηλεκτρονική διεύθυνση www.carto.net [18]. Το τμήμα αυτό της εφαρμογής αποτελεί το interface με τον χρήστη. Υλοποιείται εξ ολοκλήρου ως ένα svg και χρησιμοποιεί τεχνικές Ajax. Για την υλοποίηση της εφαρμογής χρησιμοποιούνται τα παρακάτω αρχεία:

- University.html
- UniversityFinal.svg
- button.js
- checkbox_and_radiobutton.js
- helper_functions.js
- mapApp.js
- navigation.js
- selectionList.js
- slider.js
- timer.js

Σημειώνεται ότι τα αρχεία javascript πρέπει να αποθηκευτούν στο ίδιο directory με το UniversityFinal.svg καθώς περιλαμβάνουν τις συναρτήσεις javascript που καλούνται από το τελευταίο. Το αρχείο university.html δεν είναι απαραίτητο καθώς απλά καλεί το

UniversityFinal.svg. Το σύνολο των ανωτέρω αρχείων αποθηκεύτηκαν στο directory C:\ms4w\Apache\htdocs\UniversityFinal. Λαμβάνοντας υπόψη ότι με την εγκατάσταση του maserver/apache server το αρχικό directory που «βλέπουμε» ως localhost είναι το C:\ms4w\Apache\htdocs, για να τρέξει η εφαρμογή καλούμε τη σελίδα: http://localhost/UniversityFinal/University.html. Παρακάτω αναλύεται το αρχείο UniversityFinal.svg το οποίο αποτελεί τον πυρήνα της εφαρμογής στον client.

6.3.1. Ανάλυση κώδικα svg αρχείου στον client

6.3.1.1 Σύστημα Συντεταγμένων

Στο αρχείο UniversityFinal.svg χρησιμοποιούνται τρία διαφορετικά συστήματα συντεταγμένων. Το πρώτο σχετίζεται με την οθόνη απεικόνισης (device oriented) και είναι ορισμένο σε pixels. Καθορίζεται στο root element του svg αρχείου (γραμμή 3, 1024x768) στο attribute viewbox. Στο ίδιο σημείο του κώδικα ορίζεται το πλάτος και ύψος στο 100%. Στην περίπτωση που αλλαχθεί αυτό το σύστημα συντεταγμένων της οθόνης, θα πρέπει να προσαρμοστούν αντίστοιχα και τα μεγέθη των συμβόλων, που ορίζονται στο αρχείο αυτό, προκειμένου να ανταποκρίνονται στις αλλαγές των συντεταγμένων.

Το δεύτερο σύστημα συντεταγμένων είναι αυτό του παραθύρου του κυρίως χάρτη (main map window στη γραμμή 382, στο φωλιασμένο svg στοιχείο). Το σύστημα αυτό ορίζεται σε μονάδες πραγματικών συντεταγμένων (μέτρα). Σημειώνεται ότι στο σύστημα συντεταγμένων του viewbox ο άξονας y είναι ανεστραμμένος, καθώς τα συστήματα συντεταγμένων των χαρτών λειτουργούν αντίθετα από το σύστημα συντεταγμένων του SVG. Επίσης επισημαίνεται ότι, για τη σωστή και αναλλοίωτη απεικόνιση των χαρτών, θα πρέπει η αναλογία πλάτους και ύψους του εξωτερικού mainMap να είναι ίδια με αυτή του viewbox και βέβαια όμοιο με την αντίστοιχη αναλογία του χάρτη που λαμβάνουμε από το server. Για το λόγο αυτό επιλέχθηκε width=800 και height = 344. Επισημαίνεται ότι ο χάρτης που λαμβάνεται από το server είναι διαστάσεων 86m μήκους και 37m πλάτους.

Το τρίτο σύστημα συντεταγμένων αφορά στο χάρτη αναφοράς και είναι το ίδιο με αυτό του main map, πράγμα που σημαίνει ότι το viewbox του referenceMap θα πρέπει να είναι ίδιο με αυτό του MainMap. Έτσι στη γραμμή 452 ορίζουμε το <svg id=”referenceMap” με viewbox ίδιο με αυτό του MainMap. Στην ίδια γραμμή ρυθμίζουμε το πλάτος και το ύψος του χάρτη αναφοράς καθώς και τις συντεταγμένες (x,y) όπου βρίσκεται η κάτω αριστερή γωνία του.

Στη φωλιασμένη περιοχή με id=”referenceMap” μπορούμε να εισάγουμε απλά σχήματα και εικόνες είτε ανυσματικές είτε raster. Στην περίπτωσή μας τοποθετήθηκε μια raster απεικόνιση της κάτοψης του ισογείου του πανεπιστημίου. Η εικόνα αυτή δε χρειάζεται να είναι λεπτομερής καθώς χρησιμοποιείται μόνο για να δίνει συνολική αίσθηση του χώρου στο χρήστη καθώς αυτός κάνει zoom και pan πάνω στο χάρτη (ιδιαίτερα χρήσιμο χαρακτηριστικό στις περιπτώσεις που έχει γίνει μεγάλο zoom).

Στη φωλιασμένη περιοχή με id=”mainMapGroup” εισάγουμε όλα τα στοιχεία που θέλουμε να απεικονίσουμε (ανυσματικές εικόνες, σχήματα, δυναμικά layers ή στατικές εικόνες κλπ). Όλα τα παραπάνω στοιχεία θα πρέπει να εισαχθούν μέσα σε αυτό το φωλιασμένο group προκειμένου να είναι εφικτές οι λειτουργίες pan και zoom. Τονίζεται και πάλι ότι το σύστημα συντεταγμένων είναι ανεστραμμένο ως προς τον άξονα y. Για το λόγο αυτό όλα τα layers ξεκινούν με συντεταγμένες (0,-37). Το viewbox κάθε layer έχει ύψος και πλάτος όμοιο με του εξωτερικού παραθύρου (mainMap) ενώ για κάθε svg id ορίζεται ότι θα καλύψει το 100% του width και height μέσα στο viewbox του mainMap.

6.3.1.2 Αρχικοποίηση παραμέτρων προγράμματος προβολής

Απαραίτητη προϋπόθεση για τη λειτουργία της εφαρμογής μας είναι η δήλωση των δύο καθολικών μεταβλητών myMapApp και myMainMap καθώς επίσης και ο ορισμός των μεταβλητών dynamicLayers και digiLayers στη συνάρτηση αρχικοποίησης (init) του κώδικα. Οι μεταβλητές αυτές αρχικά είναι άδειοι πίνακες, οι οποίοι χρησιμοποιούνται για να προσθέσουμε δυναμικά επίπεδα (layers). Για κάθε layer που επιθυμούμε να έχουμε, εισάγουμε και μια τιμή στον πίνακα dynamicLayers όπως φαίνεται στις γραμμές 22 έως 49. Η μεταβλητή digiLayers είναι απαραίτητο να ορισθεί, αν και δε θα χρησιμοποιηθεί. Οι πίνακες αυτοί χρησιμοποιούνται από το map αντικείμενο (mainMap) το οποίο ορίζεται στη γραμμή 59. Το αντικείμενο αυτό περιέχει διάφορα δεδομένα του χάρτη και επίσης είναι αυτό που επιτρέπει τη λειτουργία των εργαλείων zoom και pan. Ως παράμετροι του map αντικείμενου τίθενται οι εξής:

mapName (string): id του mainMap (φωλιασμένο svg)

maxWidth (number, viewbox coordinates): μέγιστο πλάτος του χάρτη όπως ορίζεται στο σύστημα συντεταγμένων του εσωτερικού viewbox (πραγματικές μονάδες)

minWidth (number, viewbox coordinates): ελάχιστο πλάτος του χάρτη όπως ορίζεται στο σύστημα συντεταγμένων του εσωτερικού viewbox (πραγματικές μονάδες)

zoomFact (number): Συντελεστής μεταξύ 0 και 1 που ορίζει το ποσοστό zoom in και zoom out κάθε φορά που πατάμε τα κουμπιά +/-. Για παράδειγμα μια τιμή 0,6 σημαίνει ότι στο επόμενο zoom in το extent του χάρτη θα είναι το 60% του προηγούμενου extent.

nrDecimals (number, integer): ο αριθμός των δεκαδικών που θα εμφανίζεται ή θα χρησιμοποιηθεί για την ψηφιοποίηση (δεν χρησιμοποιείται)

epsg (number, integer): Η epsg προβολή του χάρτη. Με δεδομένο ότι δεν χρησιμοποιούμε προβολή του χάρτη ο αριθμός αυτός μπορεί να είναι οποιοσδήποτε.

Units (string): Περιγράφει τη μονάδα μέτρησης του χάρτη (“m” για τα μέτρα που χρησιμοποιούμε εμείς)

unitsFactor (number): Συντελεστής που χρησιμοποιείται για μετατροπή συντεταγμένων όταν αυτές προβάλλονται κατά την κίνηση του mouse. Για παράδειγμα μπορούμε να τον χρησιμοποιήσουμε για μετατροπή από πόδια σε μέτρα και αντίστροφα. Τίθεται σε 1 εφόσον δε θέλουμε να χρησιμοποιήσουμε διαφορετικές συντεταγμένες.

showCoords (Boolean): true|false, Καθορίζει εάν θα χρησιμοποιείται η συνάρτηση showCoordinates για την προβολή των συντεταγμένων με την κίνηση του mouse.

coordXId (string): id του text element για την προβολή των x τιμών

coordYId (string): id του text element για την προβολή των y τιμών

dynamicLayers (Array of strings): πίνακας που περιέχει σύνολο ids των δυναμικών επιπέδων (layers) που θα φορτώνονται.

digiLayers (Array of strings): πίνακας που περιέχει ψηφιοοποιημένα επίπεδα (κενός σε εμάς)

active DigiLayer (string): το id του τρέχοντος digitizing layer (κενό σε εμάς)

zoomRectAttribs (Πίνακας που περιλαμβάνει γνωρίσματα που αφορούν τον τρόπο προβολής): Τα γνωρίσματα αυτά καθορίζουν την εμφάνιση του παραλληλογράμμου zoom που χρησιμοποιείται στο manual zooming. Το stroke-width και το stroke-dasharray ορίζονται σε σχέση με το width του υφιστάμενου map, ενώ το συντακτικό του stroke-dasharray είναι πολύ αυστηρό. Για παράδειγμα var zoomRectstyles = { “fill”:”none”,“stroke”:0.002,“stroke-dasharray”:“0.012,0.002,"stroke-dasharray" : "0.012, 0.002"}

highlightAttribs (Πίνακας που περιέχει γνωρίσματα που αφορούν τρόπο εμφάνισης): Τα γνωρίσματα αυτά αφορούν τη μορφή των croshair γραμμών για την κατάδειξη των point features. Θα πρέπει κατ’ ελάχιστο να περιλαμβάνει τις μεταβλητές stroke και stroke-

width οι οποίες ορίζονται σε σχέση με το width του υφιστάμενου map, ενώ το συντακτικό του stroke-dasharray είναι πολύ αυστηρό. Για παράδειγμα var highlightStyles = {"stroke":"crimson","stroke-width":0.002}

dragRectAttribs (Πίνακας που περιέχει γνωρίσματα που αφορούν τρόπο εμφάνισης): Τα γνωρίσματα αυτά αφορούν τη μορφή του παραλληλογράμμου που εμφανίζεται όταν κάνουμε drag στο χάρτη και δείχνει το τρέχων extent του χάρτη στον χάρτη αναφοράς. Θα πρέπει κατ' ελάχιστον να περιλαμβάνει τις μεταβλητές stroke και stroke-width οι οποίες ορίζονται σε σχέση με το width του υφιστάμενου map, ενώ το συντακτικό του stroke-dasharray είναι πολύ αυστηρό. Για παράδειγμα var dragRectStyles = {"fill":"lightskyblue","fill-opacity":0.5}.

refnapName (string): Το id του χάρτη αναφοράς (reference map).

dragSymbol (string): Το id του στοιχείου που περιγράφει το σύμβολο που εμφανίζεται όταν ο χρήστης κάνει πολύ μεγάλο zoom in ξεπερνώντας κάποιο όριο

symbolThreshold (number, viewbox units): το όριο, οριζόμενο σε μονάδες viewbox (πλάτος του παραλληλογράμμου drag). Στην περίπτωση που το drag παραλληλόγραμμο είναι μικρότερο από αυτό το όριο τότε εμφανίζεται το dragSymbol.

Στη γραμμή 63 καλείται η zoomSlider. Το slider αντικείμενο εισάγεται ως property στο αντικείμενο myMapApp ώστε να αποφευχθεί ο ορισμός περισσότερων καθολικών μεταβλητών. Για τον ορισμό του αντικειμένου slider απαιτούνται οι παρακάτω παράμετροι:

x1: x coordinate of the slider start point

y1: y coordinate of the slider start point

value1: the value of the slider start point (the minimum width at maximal zoom in)

x2: x coordinate of the slider end point

y2: y coordinate of the slider end point

value2: the value of the slider end point (the maximum width at full view)

startVal: the slider start value (in our case myMainMap.origWidth)

sliderGroupId: the id of the group where we will place the slider (in our case mapZoomSlider, part of the group containing the navigation tools)

sliderColor: the color of the slider line (defined verbally or in rgb)

visSliderWidth: the width of the slider line

invisSliderWidth: the width of an invisible slider line receiving the slider events (for very thin slider lines, the width should be considerably bigger)

sliderSymb: the id of a slider symbol (in our case "sliderSymbol")

functionToCall: the function or object to be called when the slider value was changed (in our case the "myMapApp.refMapDragger" object instance)

mouseMoveBool: a boolean flag indicating if we want instant feedback while moving the slider or no (in our case true).

Στη συνέχεια ορίζονται τα απαραίτητα για το interface με το χρήστη buttons. Τα στοιχεία αυτά χρησιμοποιούν το button object. Η σχετική με τα αντικείμενα buttons βιβλιογραφία βρίσκεται στη διεύθυνση http://www.carto.net/papers/svg/gui/button/.

Η μορφή των buttons μπορεί να τροποποιηθεί με χρήση presentation attributes ή CSS στυλ. Οι ορισμοί των συμβόλων γίνονται στα αντίστοιχα τμήματα <symbols /> (που είναι μέσα στο τμήμα <defs />). Επισημαίνεται ότι μπορούμε να μεταβάλλουμε το στυλ, τη μορφή και τη θέση των buttons αλλά όχι τις σχετικές με αυτά συναρτήσεις και τα id αυτών αλλιώς οι συναρτήσεις πλοήγησης δε θα λειτουργούν.

Στη γραμμή 66, με την κλήση της myMapApp.buttons = new Array();, δημιουργείται ένας πίνακας στο αντικείμενο myMapApp ο οποίος κρατά όλες τις αναφορές στα buttons zoom και pan.

Μετά τη δημιουργία των buttons καλούμε τη συνάρτηση checkButtons του αντικειμένου myMainMap. Η συνάρτηση αυτή αναλύει τα extents του υφιστάμενου χάρτη και ενεργοποιεί/απενεργοποιεί τα buttons. Σε μερικές περιπτώσεις τα buttons δε θα πρέπει να είναι ενεργά. Για παράδειγμα όταν ο χάρτης εμφανίζεται σε πλήρη προβολή (δεν έχει γίνει zoom), τα “zoomOut”, “panManual” και “recenterMap” είναι απενεργοποιημένα. Η υπόψη συνάρτηση καλείται μετά από κάθε μεταβολή των ορίων (extents) του χάρτη.

Τέλος στη γραμμή 109 καλείται η συνάρτηση loadProjectSpecific. Η ίδια συνάρτηση καλείται πάντα μετά από επανακαθορισμό του ορίων του χάρτη (π.χ. στις περιπτώσεις zoom, pan). Με αυτή τη συνάρτηση προσδιορίζουμε το τι πρέπει να γίνεται σε κάθε αλλαγή των ορίων του χάρτη. Στην περίπτωσή μας επαναπροσδιορίζουμε τους αριθμούς που δηλώνουν το μήκος και πλάτος του map extent και καλούμε τι συνάρτηση για τη δυναμική προβολή των επιθυμητών layers.

Περισσότερες λεπτομέρειες σχετικά με τον τρόπο λειτουργίας των ανωτέρω αντικειμένων μπορεί κάποιος να βρει στην ηλεκτρονική διεύθυνση http://www.carto.net/papers/svg/navigationTools/

6.3.1.3 Προετοιμασία του τμήματος εφαρμογής στον client για δυναμική εισαγωγή δεδομένων

Ο χάρτης ο οποίος τελικά προβάλλεται στον χρήστη αποτελείται από ένα άθροισμα επικαλυπτόμενων επιπέδων (layers). Τα επίπεδα αυτά εισάγονται δυναμικά. Υπάρχουν δύο δυνατές επιλογές για την παραγωγή και εμφάνιση των επιπέδων αυτών. Η μία είναι να ελέγχεται από το πρόγραμμα το ποια layers έχουν επιλεγεί από το χρήστη (ελέγχοντας τα αντίστοιχα checkbuttons) και να πραγματοποιείται μια κλήση στον server ζητώντας το σύνολο των επιλεγμένων κάθε φορά layers. Η άλλη επιλογή, η οποία και ακολουθήθηκε, είναι για το κάθε layer να πραγματοποιείται και μια κλήση με τις κατάλληλες παραμέτρους στο server. Όπως έχει ήδη αναφερθεί, για τις κλήσεις αυτές χρησιμοποιείται η τεχνολογία Ajax. Ένα από τα πλεονεκτήματα που παρέχει η τεχνολογία αυτή είναι και το γεγονός ότι ακόμα και εάν κάποια από τις κλήσεις για κάποιο λόγο δεν είναι δυνατόν να ολοκληρωθεί το πρόγραμμα δεν «κολλάει» περιμένοντας την απάντηση αλλά συνεχίζει να λειτουργεί (χωρίς βέβαια να απεικονίζει αποτέλεσμα από τη συγκεκριμένη κλήση). Για τον ίδιο λόγο επιλέχθηκε η πραγματοποίηση ξεχωριστών κλήσεων στο server ανά layer, έτσι ώστε αν για κάποιο λόγο ο server δεν μπορεί να δημιουργήσει κάποιο επίπεδο, να είναι δυνατή η απεικόνιση των υπολοίπων. Επιπρόσθετα με τον τρόπο αυτό, στις περιπτώσεις όπου δεν αλλάζουμε το extent του χάρτη και απλά ο χρήστης τοποθετεί ή αφαιρεί συγκεκριμένα layers από το χάρτη, ο server καλείται και επιστρέφει μόνο τα layer αυτά και όχι όλο το χάρτη εξοικονομώντας έτσι bandwidth.

Κάθε layer που επιστρέφεται από το server είναι ένα svg αρχείο, το οποίο θα πρέπει να εισαχθεί στο τμήμα της εφαρμογής που υλοποιείται στον client, προκειμένου να απεικονιστεί στο interface του χρήστη. Για το σκοπό αυτό στις γραμμές 387 – 400 ορίζουμε svg τμήματα, με IDs όμοια με αυτά των layers που θέλουμε να εισάγουμε, προκειμένου χρησιμοποιώντας javascript να κάνουμε εισαγωγή των απαντήσεων του server στις συγκεκριμένες θέσεις του svg αρχείου, το οποίο εκτελείται στον client. Με τον τρόπο αυτό μας δίνεται και η δυνατότητα να κάνουμε χρήση και μεθόδων της javascript όπως η onclick ώστε το περιεχόμενο της σελίδας μας να γίνει ακόμη πιο δυναμικό. Ας μην ξεχνάμε ότι το αποτέλεσμα που παράγεται από τον server είναι ένα svg αρχείο, το οποίο όμως δεν περιέχει συναρτήσεις όπως η onclick, onmouseover

κλπ. Εάν δεν χρησιμοποιούσαμε έναν «καμβά», όπως αυτόν στις γραμμές 387-400, η εισαγωγή των ανωτέρω συναρτήσεων θα έπρεπε να είναι στον server με χρήση συναρτήσεων php που ελέγχουν το DOM Document, κάτι που είναι αρκετά δύσχρηστο και δύσκολο.

Προκειμένου να καταστεί εφικτή η εισαγωγή των layers, που λαμβάνονται από στο server, στο svg αρχείο του client, στη συνάρτηση αρχικοποίησης (init) για κάθε ζητούμενο layer εισάγουμε και μια αντίστοιχη τιμή στον πίνακα dynamicLayers. Το γνώρισμα “key” αποτελεί το όνομα του layer, το “value” καθορίζει εάν το layer θα « φορτώνεται» ή όχι και το “loaded” καθορίζει εάν το layer έχει ήδη φορτωθεί.

Επίσης χρησιμοποιούμαι και την παράμετρο timestamp, η οποία προστίθεται στο αντικείμενο myMainMap, προκειμένου να ελέγχεται καλύτερα η διαδικασία « φόρτωσης» των δεδομένων, στις περιπτώσεις που ο χρήστης κάνει zoom ή pan πολύ γρήγορα. Για την ακρίβεια η παράμετρος αυτή χρειάζεται μόνο εάν ο χρήστης εκτελεί τις παραπάνω ενέργειες πιο γρήγορα από το χρόνο που απαιτείται για να ληφθούν από τον server τα δεδομένα που ζήτησε ο client. Με κάθε κλήση προς το server στέλνεται ως παράμετρος και η μεταβλητή timestamp (η οποία τροποποιείται με κάθε κλήση). Κάθε φορά που ο χρήστης κάνει zoom ή pan η timestamp μεταβάλλεται. Εάν ο χρήστης κάνει pan ή zoom πριν ληφθούν τα δεδομένα που ζητήθηκαν προηγουμένως τότε τα δεδομένα αυτά δεν προστίθενται στο document tree αλλά απλά αγνοούνται. Τα δεδομένα αυτά προστίθενται στο DOM tree μόνο εφόσον το timestamp τους είναι ίδιο με το τρέχων timestamp. Η timestamp δημιουργείται στις γραμμές 116-117, όπου παίρνει την τρέχουσα τιμή ημερομηνίας και ώρας (εκφρασμένη σε msec από την ημερομηνία 1/1/1970). Στη γραμμή 123 καλούμε τη συνάρτηση getLayers προκειμένου να λάβουμε τα ζητούμενα layers. Επισημαίνεται ότι η loadProjectSpecific, πέραν της αρχικής κλήσης της κατά την έναρξη της εφαρμογής, καλείται με κάθε zoom ή pan του χρήστη.

6.3.1.4 Δυναμική εισαγωγή δεδομένων

Για τη δυναμική εισαγωγή των δεδομένων (layers) χρησιμοποιούνται οι συναρτήσεις getLayers(), getSingleLayer(layerId), createXmlHttpRequestObject() και η handleResults(node).

Η συνάρτηση getLayers() εκτελεί ένα βρόχο μέσα στον οποίο ελέγχει όλα τα dynamicLayers (που είναι ένα γνώρισμα τύπου πίνακα του αντικειμένου myMainMap). Εφόσον η “value” του τρέχοντος dynamicLayer είναι “yes”, καλείται η getSingleLayer(layerId) ώστε να γίνει μια κλήση στο server για ανάκτηση του συγκεκριμένου layer. Πριν γίνει αυτό όμως ελέγχουμε εάν υπάρχει καταχώρηση για το

υφιστάμενο timestamp στον πίνακα myMainMap.nrLayerToLoad. Ο πίνακας αυτός κρατά τον αριθμό των layers που θα φορτωθούν με το τρέχον timestamp. Αν η ανωτέρω καταχώρηση δεν υπάρχει δημιουργείται, ενώ αν υπάρχει ήδη (στις περιπτώσεις που φορτώνονται πάνω από ένα layer) αυξάνεται η τιμή της. Για παράδειγμα εάν υπάρχει απαίτηση να φορτωθούν 3 layers η τιμή της myMainMap.nrLayerToLoad[myMainMap.timestamp.toString()] θα είναι 3.

Στη συνέχεια στη γραμμή 144 ελέγχουμε εάν η τιμή είναι 1 οπότε εμφανίζουμε ένα text element το οποίο γράφει «loading data», που σημαίνει ότι γίνεται κλήση στο server για ανάκτηση δεδομένων.

Η συνάρτηση getSingleLayer(layerId) λαμβάνει ως παράμετρο το όνομα του layer το οποίο θέλουμε να ανακτήσουμε από το server. Η μεταβλητή myUrlString, στη γραμμή 155, περιέχει την κλήση της συνάρτησης στον server καθώς και όλες τις απαιτούμενες παραμέτρους για την κλήση αυτή. Οι εν λόγω παράμετροι είναι:

layername: Το όνομα του layer που θέλουμε να λάβουμε από το server

timestamp: Το timestamp της κλήσης που κάνουμε, το οποίο στη συνέχεια ο server ενσωματώνει στο αποτέλεσμα που επιστρέφει, ώστε να ξέρουμε σε ποια κλήση αναφέρεται η απάντηση που λαμβάνουμε

xmin,ymin: Η τιμή (x,y) της κάτω αριστερής γωνίας του παραθύρου θέασης που έχει επιλέξει ο χρήστης. Το παράθυρο θέασης αρχικά είναι όλος ο χάρτης ενώ στις περιπτώσεις που έχει γίνει zoom είναι το τμήμα του χάρτη που ο χρήστης έχει ζητήσει να γίνει μεγέθυνση. Το παράθυρο αυτό δίδεται μέσω των γνωρισμάτων του myMainMap και είναι εκφρασμένο σε πραγματικές διαστάσεις (μέτρα). Σημειώνεται ότι το σημείο (myMainMap.curxOrig, mayMainMap. curyOrig), το οποίο εφόσον δεν έχει γίνει zoom είναι το (0,0) είναι η πάνω αριστερή γωνία του παραθύρου θέασης. Όμως το σημείο (0,0) για το layer που δημιουργεί και αναγνωρίζει ο mapserver είναι η κάτω αριστερή γωνία. Για το λόγο αυτό η μεταβλητή ymin αποστέλλεται στον server ανεστραμμένη. Για τον υπολογισμό της παίρνουμε την myMainMap.curyOrig προσθέτουμε το ύψος του παραθύρου θέασης και πολλαπλασιάζουμε επί -1.

xmax,ymax: Το σημείο πάνω δεξιά του παραθύρου θέασης. Για τον υπολογισμό του xmax προσθέτουμε στο myMainMap.curxOrig το πλάτος του παραθύρου θέασης και για το ymax απλά πολλαπλασιάζουμε επί -1 το myMainMap.curyOrig.

Από τη γραμμή 156 ξεκινά η κλήση προς το server χρησιμοποιώντας τεχνολογία Ajax.

6.3.1.4.1 Κλήσεις Server με χρήση AJAX

Στη γραμμή 156 καλείται η συνάρτηση createXmlHttpRequestObject() η οποία ορίζεται στη γραμμή 189.

```
function createXmlHttpRequestObject()
  {//αποθήκευση του δείκτη στο αντικείμενο XMLHttpRequest
   var xmlHttp;
   //για όλους τους browsers εκτός του IE6 και παλιότερων
   try
   {//δημιουργία του XMLHttpRequest αντικειμένου
    xmlHttp = new XMLHttpRequest();
   }
   catch(e)
   {//σε περίπτωση του IE6 και παλιότερων
    var XmlHttpVersions = new Array("MSXML2.XMLHTTP.6.0",
    "MSXML2.XMLHTTP.5.0",
    "MSXML2.XMLHTTP.4.0",
    "MSXML2.XMLHTTP.3.0",
    "MSXML2.XMLHTTP",
    "Microsoft.XMLHTTP");
    //δοκιμάζω κάθε version έως ότου κάποια δουλέψει
    for (var i=0; i<XmlHttpVersions.length && !xmlHttp; i++)
    {
     try
     {//δοκίμασε να δημιουργήσεις το αντικείμενο XMLHttpRequest
      xmlHttp = new ActiveXObject(XmlHttpVersions[i]);
     }
     catch (e) {}
```

```
    }
  } // επιστροφή του αντικειμένου που δημιουργήθηκε ή μήνυμα λάθους
  if (!xmlHttp)
    alert("Error creating the XMLHttpRequest object.");
  else
    return xmlHttp;
}
```

Η ανωτέρω συνάρτηση έχει δημιουργηθεί έτσι ώστε να μπορεί να λειτουργήσει ανεξαρτήτως browser. Στον Internet Explorer 6 και σε παλαιότερες από αυτόν εκδόσεις η XMLHttpRequest υλοποιείται ως ActiveX control το οποίο αρχικοποιείται ως εξής:

xmlhttp= new ActiveXObject (“Microsoft.XMLHttp”);

Για όλους του άλλους browsers η XMLHttpRequest είναι γηγενές (native) αντικείμενο οπότε μπορούμε να το δημιουργήσουμε ως εξής:

xmlhttp= new XMLHttpRequest();

Χρησιμοποιώντας τη δομή try/catch δημιουργούμε το αντικείμενο XMLHttpRequest() καλώντας την new XMLHttpRequest(); και σε περίπτωση που η Javascript επιστρέψει error (exception), το οποίο συμβαίνει όταν ο browser είναι IE 6 ή παλαιότερης έκδοσης, οδηγούμαστε στο τμήμα του κώδικα που θα χειριστεί το exception αυτό (τμήμα catch). Σε περίπτωση που δεν προκύψει exception το τμήμα catch δε θα εκτελεστεί ποτέ. Σε περίπτωση που απαιτηθεί να χειριστούμε το exception απλά δοκιμάζουμε σε κάθε version του IE να δημιουργήσουμε το αντικείμενο XMLHttpRequest(). Τελικά στη γραμμή 220 ελέγχουμε για την επιτυχή δημιουργία του αντικειμένου αυτού και είτε επιστρέφεται μήνυμα σφάλματος είτε επιστρέφεται το ίδιο το αντικείμενο.

Το παραπάνω αντικείμενο αποθηκεύεται στη μεταβλητή xmlHttp. Στη συνέχεια στη γραμμή 157 καλείται η συνάρτηση open του αντικειμένου xmlHttp. Σε κάθε κλήση προς το server χρησιμοποιούνται οι συναρτήσεις open και send. Η open αρχικοποιεί/διαμορφώνει την κλήση θέτοντας διάφορες παραμέτρους και η send εκτελεί την κλήση (αποκτά πρόσβαση στο server). Όταν η κλήση γίνεται ασύγχρονα όπως στην περίπτωσή μας, πριν από την κλήση της send απαιτείται να αρχικοποιήσουμε το γνώρισμα onreadystatechange με μια συνάρτηση επιστροφής (callback method) η οποία εκτελείται όταν η κατάσταση (status) της κλήσης (request) προς το server αλλάξει. Η συνάρτηση αυτή σε εμάς είναι η handleHttpReturnedData() που ουσιαστικά

λαμβάνει το svg αρχείο από το server. Η συνάρτηση open δέχεται απαραιτήτως ως ορίσματα δύο παραμέτρους ενώ προαιρετικά μπορεί να δεχθεί και επιπλέον ορίσματα. Τονίζεται ότι η open δεν ξεκινά τη σύνδεση με το server αλλά θέτει απλά τις παραμέτρους της σύνδεσης. Η πρώτη παράμετρος που δέχεται προσδιορίζει τη μέθοδο που θα χρησιμοποιηθεί για την αποστολή δεδομένων στο server. Η μέθοδος αυτή μπορεί να είναι μία εκ των GET, PUT ή POST. Η μέθοδος POST είναι απαραίτητη στην περίπτωση που στέλνονται δεδομένα χωρητικότητας μεγαλύτερης των 512 bytes. Η δεύτερη παράμετρος είναι η URL, η οποία καθορίζει που θα σταλεί η κλήση. Η URL μπορεί να είναι απόλυτη / πλήρης ή σχετική. Η τρίτη παράμετρος της open ονομάζεται async και καθορίζει εάν η κλήση είναι ασύγχρονη. Εφόσον οριστεί ως true σημαίνει ότι το πρόγραμμα θα συνεχίσει κανονικά την εκτέλεσή του μετά την κλήση της send χωρίς να περιμένει την απάντηση (response) από το server. Εφόσον η τιμή αυτή οριστεί ως false τότε το πρόγραμμα (η ιστοσελίδα) θα παγώσει έως ότου λάβει την απάντηση από το server. Στην περίπτωσή μας ζητάμε να εκτελεστεί η λειτουργία “get” με url όπως έχει αναφερθεί νωρίτερα και τα δεδομένα να ληφθούν από το server ασύγχρονα.

Όπως προαναφέρθηκε μετά την κλήση της send στη γραμμή 159 το πρόγραμμα δεν παγώνει αλλά συνεχίζει να εκτελείται κανονικά. Η συνάρτηση handleHttpReturnedData() αναλαμβάνει να χειριστεί τις αλλαγές της κατάσταση κλήσης. Η συνάρτηση αυτή καλείται 4 φορές, μία για κάθε αλλαγή κατάστασης. Σημειώνεται ότι το γνώρισμα readyState του αντικειμένου xmlHttprequest παίρνει τις παρακάτω τιμές:

- 0 = unintialized
- 1 = loading
- 2 = loaded
- 3 = interactive
- 4 = complete

Η κατάσταση interactive είναι μια ενδιάμεση κατάσταση όταν η απάντηση (response) έχει μερικώς ληφθεί. Οι άλλες καταστάσεις είναι προφανείς. Στην εφαρμογή μας χρησιμοποιούμε μόνο την κατάσταση 4 που σημαίνει ότι η απάντηση από το server έχει ληφθεί.

Για το σκοπό αυτό στη γραμμή 162 ελέγχουμε εάν είμαστε στην κατάσταση 4 και στη συνέχεια εφόσον το status της Http είναι «OK» (τιμή 200) λαμβάνουμε την απάντηση του server στη μεταβλητή xmlResponse. Σημειώνεται ότι, δεδομένου ότι το svg αρχείο είναι ένα xml αρχείο, επιλέξαμε να χρησιμοποιήσουμε τη συνάρτηση

xmlHttp.responseXML και όχι την xmlHttp.responseText. Επίσης αναφέρεται ότι ακριβώς επειδή θέλουμε να λάβουμε το svg αρχείο κατευθείαν από το server στη μορφή xml ορίσαμε και στην php στο server ότι η απάντηση που θα αποστείλει θα είναι xml (header('content-Type: image/svg+xml);), ενώ το output από το server αποθηκεύεται με τη συνάρτηση $output=$doc->saveXML(); και επιστρέφεται στον client με την echo $output;. Το αποτέλεσμα είναι στον client να παίρνουμε την απάντηση του server ως ένα DOM document.

Στις γραμμές 169-174 γίνεται, ανάλογα με τον browser που χρησιμοποιούμε, έλεγχος του αποτελέσματος για τυχόν σφάλματα στη δομή του και στην περίπτωση που το DOM document είναι ορθά δομημένο ως XML αρχείο, το αποθηκεύουμε στη μεταβλητή xmlRoot, η οποία πλέον κρατά το root του svg αρχείου. Τέλος καλείται η handleResults(node) η οποία πλέον θα επεξεργαστεί αυτό το svg αρχείο, εντάσσοντας το στην κατάλληλη θέση στη σελίδα του client. Στην περίπτωση που δεν ληφθεί απάντηση από τον server τότε στις γραμμές 180-184 επιστρέφεται ως μήνυμα λάθους το «αδυναμία σύνδεσης με το server.»

6.3.1.4.2 Εισαγωγή των δεδομένων του server στην εφαρμογή

Με τη συνάρτηση handleResults(node) αρχικά αποθηκεύουμε στη μεταβλητή currentDynamicLayer το όνομα του layer που έχουμε λάβει, ενώ στη μεταβλητή timestamp την τιμή timestamp αυτού. Η τιμή αυτή συγκρίνεται με το τρέχον timestamp. Στην περίπτωση που οι τιμές δεν συμφωνούν μεταξύ τους (που σημαίνει ότι ο χρήστης κάνει zoom ή pan πιο γρήγορα από ότι λαμβάνει δεδομένα από το server) το layer που απεστάλη από το server αγνοείται. Στην περίπτωση που αυτές οι τιμές συμφωνούν θα πρέπει τα δεδομένα που παραλήφθηκαν από το server να ενσωματωθούν στο svg interface του client. Για να γίνει αυτό βρίσκουμε, στη γραμμή 237, με βάση το όνομα του layer τη θέση στην οποία το νέο layer θα γίνει append. Στη γραμμές 384-402 όπως έχει ήδη αναφερθεί έχουμε δημιουργήσει τον καμβά μέσα στον οποίο τοποθετούνται οι απαντήσεις από το server. Ας πάρουμε για παράδειγμα ότι ζητήθηκε το layer με id = "grCor". Μετά την εκτέλεση της γραμμής 237 γνωρίζουμε ότι το layer αυτό θα μπει κάτω από το < svg id="grCor" … >. Έτσι αν το κάναμε απλά append και εξετάζαμε το νέο DOM Document που θα προέκυπτε θα βλέπαμε κάτι σαν αυτό:

<svg id="grCor" …. >

<svg id="grCor" … >

</svg>

</svg>

Μετά από διαδοχικές κλήσεις κάνοντας zoom, pan κλπ και δεδομένου ότι κάθε φορά θα λαμβάναμε και ένα svg αρχείο με id “grCor”, το DOM Document που θα βλέπαμε θα ήταν:

<svg id=”grCor” …. >

<svg id=”grCor” … >

<svg id=”grCor” … >

<svg id=”grCor” … >

<svg id=”grCor” … >

…

</svg>

</svg>

…

</svg>

Από τα παραπάνω καθίσταται σαφές ότι θα πρέπει, πριν εισάγουμε το layer που ζητήσαμε στην θέση του στο DOM document, να ελέγχουμε εάν ήδη υπάρχει κάτι εκεί και να το διαγράφουμε. Αυτό ακριβώς γίνεται στις γραμμές 242-250.

```
if (myMainMap.dynamicLayers[currentDynamicLayer].loaded=="yes")
    {
     if (startingNodetoAppend.firstChild)
     {
       previousDataToremove=startingNodetoAppend.firstChild;
       startingNodetoAppend.removeChild(previousDataToremove);
     }
    }
```

Μετά τον παραπάνω έλεγχο και τη διαγραφή προηγούμενων svg τμημάτων εισάγουμε τα νέα δεδομένα στη θέση που πρέπει στο DOM document (γραμμή 252). Οι επόμενες γραμμές (254-255) ουσιαστικά «ευθυγραμμίζουν» την αρχή των αξόνων του svg τμήματος που εισαγάγαμε με την αρχή των αξόνων του παραθύρου θέασης, έτσι ώστε

να μπορούν τα δεδομένα που λάβαμε να απεικονιστούν στο παράθυρο θέασης που έχει επιλέξει ο χρήστης (έπειτα από κάποιο zoom για παράδειγμα). Στη συνέχεια θέτουμε την παράμετρο loaded σε “yes” προκειμένου να δείξουμε ότι το συγκεκριμένο layer έχει απεικονισθεί. Τέλος, η μεταβλητή myMainMap.nrLayerToLoad [myMainMap.timestamp .toString()] μειώνεται και όταν φθάσει στην τιμή 0 θέτουμε την παράμετρο visibility του text element “loading data” σε off. Αυτό σημαίνει ότι όλα τα ζητηθέντα layers έχουν φορτωθεί. Επισημαίνεται ότι λόγω των ασύγχρονων κλήσεων προς το server δεν υπάρχει καμία εγγύηση ότι τα δεδομένα επιστρέφουν από το server με την ίδια σειρά με την οποία πραγματοποιήθηκαν και οι κλήσεις.

Προκειμένου να ενημερωθεί ο χρήστης ότι τα δεδομένα που ζήτησε φορτώνονται από το server προστέθηκε το group element με id “loadingData”, το οποίο εμφανίζεται όταν φορτώνονται τα δεδομένα και παύει να εμφανίζεται όταν φορτωθούν όλα τα δεδομένα. Το στοιχείο αυτό δημιουργείται στις γραμμές 405-409 όπως φαίνεται παρακάτω:

```
<!-- Loading Data Status -->
 <g id="loadingData" visibility="hidden">
  <rect x="200" y="300" width="150" height="35" fill="white" fill-opacity="0.8"/>
  <text x="275" y="325" font-family="sans-serif" fill="dimgray" font-size="18px" font-weight="bold" text-anchor="middle">Loading Data ...</text>
 </g>
```

6.3.1.4.3 Προσθήκη χαρακτηριστικών επιλογής layers στην εφαρμογή

Προκειμένου να μπορεί ο χρήστης να ελέγχει ποια layers θα εμφανίζονται κάθε φορά χρησιμοποιούμε checkboxes και τη συνάρτηση toggleMapLayer (id, checkStatus, labelText). Για κάθε layer προσθέτουμε και ένα checkbox στη συνάρτηση init. Πληροφορίες για τη βιβλιοθήκη των checkboxes βρίσκει κανείς στη διεύθυνση http://www.carto.net/papers/svg/gui/checkbox_and_radiobutton/. Ακόμη προσθέτουμε και ένα checkbox για την εμφάνιση του layer της διαδρομής που έχει επιλέξει ο χρήστης να ακολουθήσει στο χάρτη. Επιλέγοντας αυτό το checkbox ουσιαστικά πληροφορούμε τον mapserver να διαβάσει το αντίστοιχο text αρχείο που περιλαμβάνει τα σημεία από τα οποία απαρτίζεται η διαδρομή και να φτιάξει το σχετικό layer.

Επιπρόσθετα χρησιμοποιούμε και τη συνάρτηση toggleMapLayer(id,checkStatus, labelText) με την οποία θέτουμε το visibility ενός layer σε on ή off.

Ως ένα τελευταίο χαρακτηριστικό έχουμε προσθέσει τη συνάρτηση

```
function msclick(evt)
  {
    sth=evt.target;
    alert (sth.getAttribute("id"));
  }
```

Η συνάρτηση αυτή σχετίζεται με την onclick συνάρτηση που έχουμε ορίσει στον καμβά που έχουμε για τη δυναμική εισαγωγή των layers και χρησιμοποιείται προκειμένου να μας εμφανίζει το id του σχήματος πάνω στο χάρτη στο οποίο ο χρήστης κάνει click. Ουσιαστικά η συνάρτηση αυτή εισήχθηκε για να δείξει ότι μπορούμε να αυξήσουμε τις δυνατότητες της εφαρμογής προσθέτοντας επιπλέον δυνατότητες που η javascript μας παρέχει (π.χ. tooltips, onmouseover κλπ).

Εν κατακλείδι το σύνολο των αρχείων με τις αντίστοιχες διαδρομές τους καταγράφονται στον παρακάτω Πίνακας 4:

Πίνακας 4: Αρχεία εφαρμογής και διαδρομές τους

Αρχείο	Διαδρομή	Σχόλια
University.html	C:\ms4w\Apache\htdocs\UniversityFinal	
UniversityFinal.svg	C:\ms4w\Apache\htdocs\UniversityFinal	Αρχείο client
button.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
checkbox_and_radiobutton.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
helper_functions.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
mapApp.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
navigation.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client

selectionList.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
slider.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
timer.js	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο client
getUniversityMap.php	C:\ms4w\Apache\htdocs\UniversityFinal	Αρχείο server
UniversityFinal.map	C:\ms4w\Apache\htdocs\UniversityFinal	Βοηθητικό αρχείο mapserver/server
universitysmall.png	C:\ms4w\Apache\htdocs\UniversityFinal	Εικόνα αναφοράς στον client
path.txt	C:\ms4w\Apache\htdocs\UniversityFinal	Αρχείο εισαγωγής διαδρομής (route) στο χάρτη
fontset.txt	C:\ms4w\Apache\htdocs	Ορισμός γραμματοσειρών για το mapserver
arial.ttf	C:\ms4w\Apache\htdocs	Γραμματοσειρά
arialbd.ttf	C:\ms4w\Apache\htdocs	Γραμματοσειρά
Αρχεία Shapefiles κάτοψης	C:\Mapdata\finalUniversityData\Ground	Δεδομένα εισόδου στο mapserver

ΟΡΟΛΟΓΙΑ

Database Management System = Σύστημα Διαχείρισης Βάσεων Δεδομένων

Descriptive Attributes = Περιγραφικές Ιδιότητες

Geographic Information System = Σύστημα Γεωγραφικών Πληροφοριών.

Global Positioning System = Παγκόσμιο Σύστημα Εντοπισμού

Indoor Navigation = Πλοήγηση σε εσωτερικούς χώρους

Location Based Services = Υπηρεσίες Βασισμένες στη Θέση

New Information and Communication Technologies = Καινούριες Τεχνολογίες Επικοινωνιών και Πληροφοριών

Navigation Services = Υπηρεσίες Πλοήγησης

Object Oriented Database = Αντικειμενοστρεφής Βάση Δεδομένων

Points of Interest = Σημεία Ενδιαφέροντος

Relational Database = Σχεσιακή Βάση Δεδομένων

Spatial Objects = Χωρικά Αντικείμενα

Universal Transverse Mercator = Παγκόσμια Εγκάρσια Μερκατορική

Vector Graphics = Ανυσματικά Γραφικά

ΑΚΡΩΝΥΜΙΑ

LBS	Location Based Services
GIS	Geographic Information System
GPS	Global Positioning System
POI	Points of Interest
UTM	Universal Transverse Mercator
DBMS	Database Management System
NICTS	New Information and Communication Technologies
OBU	On-Board Units
OGC	Open Geospatial Consortium
LOC	Open Mobile Alliance's Location Working Group
ISO	International Standard Organization
OpenLS	Open Location Services
WFS	Web Feature Service
WMS	Web Map Service
GML	Geography Markup Language
DBMS	Database Management System
WLAN	Wireless LAN
UWB	Ultra Wide Band
SVG	Scalable Vector Graphics
UTM	Universal Transverse Mercator

ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

1. (OGC), O.G.C., *Open Location Services 1.1*. 2005.

2. Shiode, N., et al., The impact and penetration of Location Based Services. 2004, CRC Press.

3. Espinoza, F., et al., GeoNotes: Social and Navigational Aspects of Location-Based Information Systems, in Ubiquitous Computing, International Conference. 2001: Atlanta, Georgia. p. 2-17.

4. Dru, M.-A. and S. Saada, *Location-based mobile services: The essentials*. 2001, Alcatel Telecommunications Review. p. 71-76.

5. Kupper, A., ed. Location-based Services Fundamentals and Operation. 2005, John Wiley & Sons Ltd.

6. Fremuth, N., A. Tasch, and M. Fränkle. Mobile Communities – new business opportunities for mobile network operators? in The 8th International Workshop on Mobile Multimedia Communications, (MoMuc'03). 2003. Munich, Germany.

7. Kropla, B., ed. Beginning MapServer: Open Source GIS Development. 2005, Apress: N.York.

8. Tyler, M., *Web Mapping Illustrated*. 2005, O'Reilly.

9. *Mapserver*. [cited; Available from: http://mapserver.gis.umn.edu/.

10. *Mapobjects*. [cited; Available from: http://umn.mapserver.ch/index_en.php.

11. *Maptools*. [cited; Available from: http://maptools.org/ms4w/.

12. Darie, C., et al., eds. *AJAX and PHP Building Responsive Web Applications*. 2006, Packt Publishing: Birmingham, UK.

13. Babin, L., ed. Beginning Ajax with PHP From Novice to Professional. 2007, Apress: N.York.

14. Richards, R., ed. *Pro PHP XML and Web Services*. 2006, Apress: N.York.

15. Herrington, J. (2005) *PHP Hacks*.

16. Heilmann, C., ed. Beginning JavaScript with DOM Scripting and Ajax From Novice to Professional. 2006, Apress: N.York.

17. Eisenberg, J.D., ed. *SVG Essentials*. 2002, O'Reilly.

18. *A place for cartography on the internet*. [cited; Available from: http://www.carto.net/.