Eötvös Loránd Tudományegyetem
Informatikai Kar
Térképtudományi és Geoinformatikai Tanszék

Tematikus térképek létrehozásának bemutatása a MapServer segítségével Bulgária példáján keresztül

Való Adrienn
térképész szakos hallgató

Témavezető:

Dr. Gede Mátyás
adjunktus

Budapest, 2012

Tartalomjegyzék

I. Bevezetés

Napjainkban egy weboldal elkészítéséhez nincs feltétlenül szükség szakember segítségére. Az emberek létrehozzák saját weboldalukat és különböző állományokkal; szöveggel, képekkel, videókkal vagy akár térképekkel töltik meg azokat. Az emberek információéhsége hatalmas és egyre szélesebb körben válnak elérhetővé ingyenes adatok, adatbázisok, de ezt a nagy mennyiségű információt könnyebb befogadni, ha nem száraz, vég nélküli adatsorok formájában állnak rendelkezésre.

A tematikus térképek közzététele igen alkalmas adatközlésre, hiszen rengeteg információt hordoznak, főként grafikus formában.

Diplomamunkámban ismertetem, hogy hogyan kapcsolhatók össze adatbázisok a webes felhasználói felülettel és hogyan készíthetők belőlük térképek egy ingyenesen hozzáférhető program, a MapServer segítségével. Bulgáriára vonatkozó – az interneten költségmentesen elérhető – statisztikai adatok segítségével bemutatom, hogy milyen típusú tematikus rétegek létrehozása lehetséges a programmal.

A célterület bemutatása után ismertetem a rendelkezésre álló adatokat, illetve hogy milyen módon készíthető belőlük felhasználásra alkalmas adatbázis. A MapServer eszköz rövid leírása után annak használatára térek rá; a különböző térképi rétegek definiálásának bemutatása után, az ismereteket felhasználva elkészítem a Bulgáriára vonatkozó általános, illetve tematikus tartalommal bíró rétegeket.

A szakdolgozat végén bemutatom, hogy az elkészült állományból hogyan tudunk interaktív térképet létrehozni és azt egy weboldalba ágyazni. Ezt az OpenLayers JavaScript könyvtárának segítségével teszem meg. A dolgozat keretén belül elkészítek egy weboldalt, melyen Bulgária interaktív térképe található különböző ki-be kapcsolható tematikus tartalmú és nem tematikus tartalmú rétegekkel.

A diplomamunkához tartozó CD-melléklet tartalmazza az általam felhasznált adatokat, az elkészített mapfile-t, illetve a weboldal html-dokumentumát.

II. Célterület

Bulgária a Balkán szívében fekszik, a Fekete-tenger partján; Magyarországtól Románia illetve Szerbia választja el, délkeleti irányban. Az összesen 1808 km hosszú országhatárán öt országgal osztozik: Románia, Szerbia, Macedónia, Görögország és Törökország. Kelet felől a Fekete-tenger 354 km hosszan határolja.

II.1. Bulgária statisztikai felosztása

Bulgária közigazgatási felosztása nem mindig volt a maival megegyező. 1999 óta a közigazgatás elsődleges szintje a megye (*oblaszt*/област). A történelem során nem csak a közigazgatási egységek felosztása, hanem nevük is változott.

A bolgár állam – a harmadik bolgár állam – megalakulása után (1878) a kerület vált a közigazgatás fő szintjévé. A kerületek ekkori számát az 1901-es közigazgatási reform során csökkentették huszonhatról tizenkettőre (Burgaszi, Sumeni, Kjusztendili, Pleveni, Plovdivi, Ruszei, Szófiai, Sztara Zagorai, Tărnovói, Várnai, Vidini, Vracai); ez egy újabb átszervezésig, 1934-ig így is maradt. A közigazgatási egységek neve is ekkor változott oblasztra, melyből hetet hoztak létre (Szófiai, Plovdivi, Sztara Zagorai, Burgaszi, Sumeni, Pleveni, Vracai). Számuk 1944-ben kettővel nőtt (Gorna Dzsumajai és Ruszei).

1949-ben az oblasztok neve ismét kerületre változott és számuk a Gorna Orjahovicai, Sumeni és Haszkovói kerületek megalakulásával tizenkettőre nőtt. Az 1950-es években újabb átszervezés következett be, melynek során huszonnyolc kerületet és kerületi jogú várost hoztak létre. Ezen kerületek a maival szinte teljesen megegyeztek. Fontos szerepet töltöttek be a tervgazdálkodásban; a népgazdasági terveket ezen a szinten dolgozták ki. A járásokat megszűntették; ezzel a lépéssel az addigi háromszintű közigazgatási rendszer kétszintűvé vált.

1987 augusztusától új közigazgatási beosztás lépett életbe, mely során a cél termelési komplexumok létrehozása volt, mely az állami támogatási rendszer területi elosztásának alapjául is szolgál. Így jött létre a kerületek összevonásából kilenc megye (oblaszt) (Szófia város, Szófia, Plovdiv, Haszkovo, Burgasz, Várna, Razgrád, Pleven és Mihajlovgrád megye).

Ezek után már csak egy nagyobb változás következett be a rendszerváltással, mikor is az 1987 előtti, huszonnyolc egységes felosztáshoz tértek vissza némi területi változásokkal (két esetben névváltoztatás is történt) viszont az egységek elnevezése továbbra is oblaszt

maradt. A mai megyék tehát a következők: Blagoevgrad, Burgasz, Dobrics, Gabrovo, Haszkovo, Kardzsali, Jambol, Kjusztendil, Lovecs, Montana, Pazardzsik, Pernik, Pleven, Plovdiv, Razgrad, Rusze, Sumen, Szilisztra, Szliven, Szmoljan, Szófia, Sztara Zagora, Targoviste, Várna, Veliko Tarnovo, Vidin és Vraca megyék illetve Szófia város.

Ezek a megyék az EUROSTAT (Európai Unió Statisztikai Hivatala) hierarchiájában NUTS 3-as szintnek felelnek meg, de kialakítottak NUTS 2-es szintű régiókat - melyek a hagyományos gazdasági nagykörzeteknek felelnek meg -, illetve NUTS 1-es szintűeket is.

1. ábra: *Bulgária felosztása statisztikai egységekre*

A megyék kistérségekre oszthatók (melyek az EU LAU-1-es szintjének felelnek meg).

III. Felhasznált adatok

III.1. Statisztikai adatok

A weboldalon megjelenő tematikus térképekhez felhasznált statisztikai adatok a bolgár Nemzeti Statisztikai Intézet (НАЦИОНАЛЕН СТАТИСТИЧЕСКИ ИНСТИТУТ) adattárából valók és a 2010-es évre vonatkoznak. Az intézet honlapjának angol nyelvű változatában a bolgár *oblaszt* (област), vagyis *megye* elnevezés *kerület*nek van lefordítva, így ezek után én is ezt a megnevezést fogom használni, de fontos leszögezni, hogy ezek a kerületek pontosan megegyeznek a korábban felsorolt megyékkel.

A különböző adatsorok mind letölthetők a weboldalról .xls formátumban, ami jó, mert könnyen kezelhető.

III.2. Térképi adatok

A MapServer számos különbözőféle vektoros adattal tud dolgozni, néhány ezek közül: ArcInfo, ESRI Shapefiles, KML, MapInfo, MySQL, WFS.

Az adatok mindegyikféle típusa egy adatforrásból és egy vagy több rétegből áll össze. Az adatforrás lényegében egy rétegcsoport, ami lehet egy fájl, ami több rétegből áll, vagy egy fájlcsoport (mappa). A réteg az adatok olyan csoportja, amely egyféle típusú vektoros formátumú adatokról tartalmaz információkat: pontról (point), törtvonalról (polyline) vagy sokszögről (polygon). Három típusát különböztetjük meg az adatformátumoknak:

1. Fájl-alapú adatok
2. Könyvtár-alapú adatok
3. Adatbázis-kapcsolatok

Számomra az adatok Shapefile-ként állnak rendelkezésre, mely ez első csoportba tartozik. Az ESRI Shapefile – a legtöbb ehhez a típushoz tartozó fájlformátummal szemben – több (legalább három) fájlból áll, melyeknek különböző kiterjesztésük van. Kötelező kiterjesztésű fájlok:

.shp – shape formátum, magát a jellemző geometriát adja meg

.shx – shape index formátum, lehetővé teszi a gyors keresést a shapefájlban

.dbf – attribútum formátum, minden alakzat attribútumát tárolja dBase formátumban

Nem kötelező kiterjesztésű fájlok sokfélék lehetnek, jelen esetben egyet használunk még:

.prj – projection formátum, egy egyszerű szöveges formátum, mely vetületi illetve a földrajzi koordináta-rendszerre vonatkozó információkat tartalmaz

III.2.1. A Shapefile

A Shapefile-ra szükségem van a tematikus térkép alaptérképének a megjelenítéséhez. Azonban én különböző adatokat is szeretnék ábrázolni, így számomra az lenne a kedvező, ha az adatforrásom a geometriai információkon kívül egyéb adatokat is tartalmazna.

A vektoros és a statisztikai adatok összekapcsolásához a MapInfo Professional programot használom illetve a MapInfo TAB formátumot. A Shapefile-hoz hasonlóan ez is több különböző kiterjesztésű fájlból tevődik össze, melyek mind szükségesek a megjelenítéshez. Egy egyszerű böngésző nézethez a következők szükségesek:

.dat – az attribútum-adatokat tartalmazó fájl, adatfájl

.tab – ASCII fájl, mely információkat tartalmaz az adatfájlról és kapcsolatot teremt más fájlokkal

Ahhoz, hogy térképet tudjunk megjeleníteni a következőkre van még szükségünk:

.map – grafikus és földrajzi információkat tartalmaz, melyek a képernyőn való megjelenítéshez szükségesek

.id – a grafikus és egyéb adatok összekapcsolásához szükséges információkat tartalmaz

.ind – indexfájl táblázatban tárolt adatokhoz, ha a mezők indexelve vannak

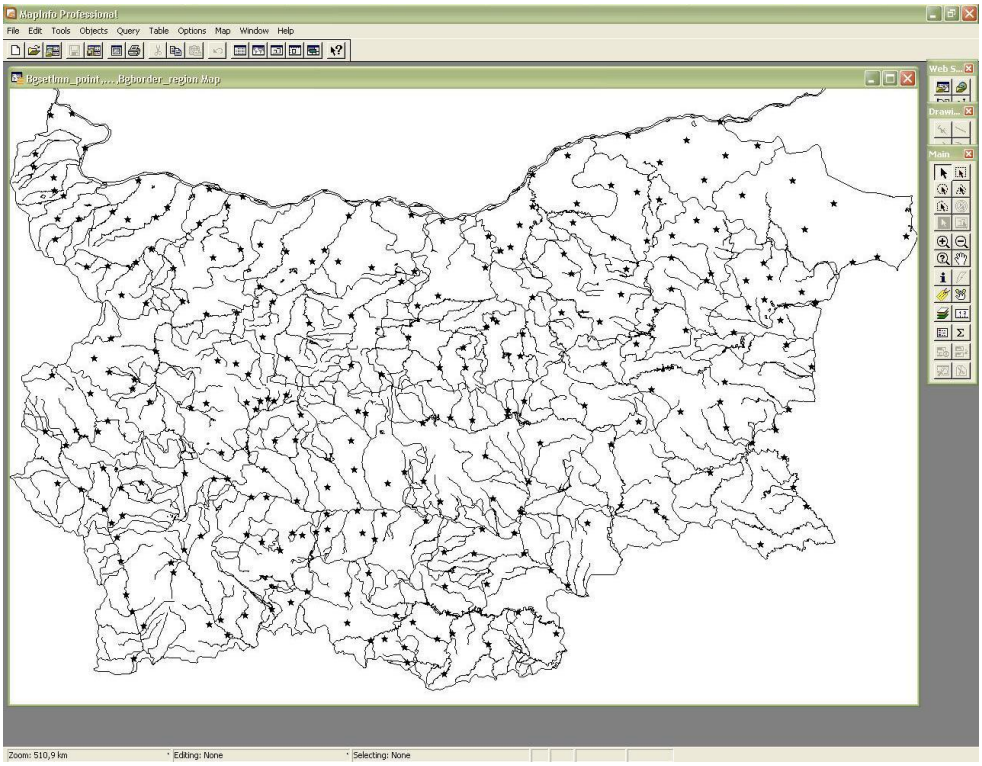
Amikor a Shapefile-t a MapInfóban meg szeretném nyitni, egyből felugrik egy párbeszédpanel, melyben elmenthetem az állományt .tab formátumba. Létrehozok egy mappát ezeknek az állományoknak és az átláthatóság kedvéért azonos néven mentem el, mint a Shapefile neve.

A következő lépésben a Shapefile-ra vonatkozó információkat állíthatok be; karakterállományt választhatok, illetve beállíthatom a vetületet, valamint a megjelenítés stílusát: a vonal típusát, vastagságát, színét, a kitöltés milyenségét, stb. Itt meghagyom az eredeti beállításokat, mert a karakterállomány és a vetület már adott . A stílus változtatása pedig nem szükségszerű.

2. ábra: *A Shapefile-ra vonatkozó információk*

A beállításokat jóváhagyva megjelenik a térképen az adat.

Minden rendelkezésemre álló Shapefile-lal elvégzem a leírt lépéseket. A művelet után egy olyan térképet kapok, melyen az ország területe, a kerületek, a tavak és a Duna sokszögként; a többi folyó törtvonalként; a városok pedig pontként jelennek meg.



3. ábra: *A Shapefile-okból létrehozott térkép*

A sokszög (polygon) típusú adatoknál érdemes figyelni arra, hogy van-e kitöltés vagy nincs, mert a rétegek - a sorrendtől függően - kitakarhatják egymást.

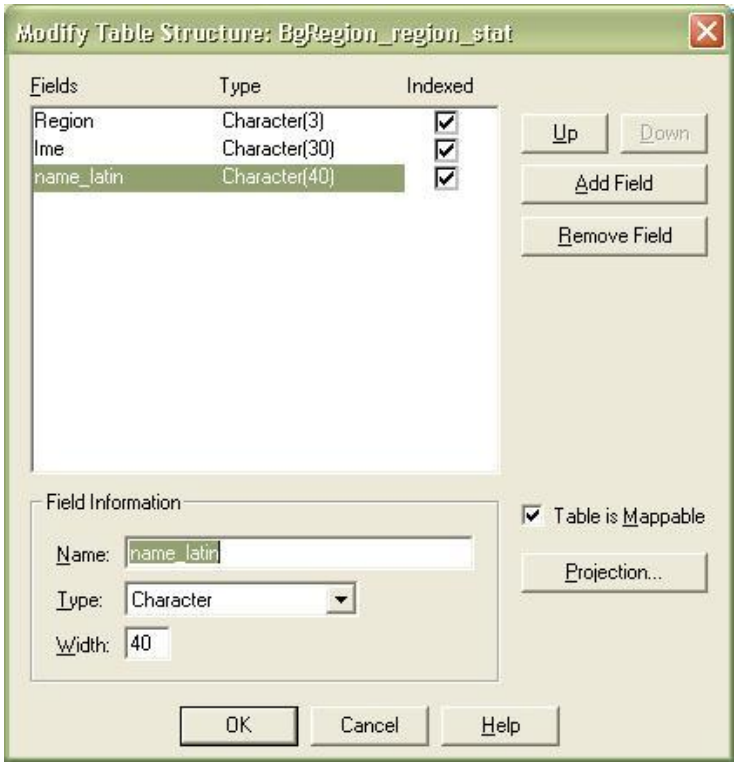
III.2.2. Adatbázis létrehozása

A következő feladatunk az adatok hozzákapcsolása a térkép objektumaihoz, pontosabban a kerületekhez.

Először el kell mentenünk a régiókat tartalmazó .tab fájlról egy másolatot a *File/Save Copy As…* parancs segítségével – mert csak ekkor lesz szerkeszthető a réteg –, majd pedig az elmentett másolatot hozzányitni a térképhez. Ezután a *Window/New Browser Window…* parancs segítségével megnyitjuk a fájlhoz tartozó *Browser* ablakot. Ez egy táblázat, mely tartalmazza az adott .tab fájlban található objektumokat, azok azonosítóját (nevét), és adott esetben a hozzájuk kapcsolódó adatokat. A táblázathoz tehát tetszőleges számú oszlop (és sor) adható, különböző típusú adatokkal.

Láthatjuk, hogy a táblázatban megjelenő eredetileg cirill betűs nevek olvashatatlanok, ezt könnyen orvosolhatjuk, ha az *Options/Text Style…* parancsra kattintva valamilyen cirill betűkészletet választunk. Emellett még láthatjuk azt is, hogy a táblázatunk eléggé üres; elsőként célszerű egy olyan oszlopot hozzáadni, ami a kerületek nevét latin betűkkel megírva tartalmazza. Az angol átírást praktikus választanunk, mivel a bolgár Nemzeti Statisztika Intézet honlapjáról letöltött adatok is angolul állnak rendelkezésünkre és a későbbiekben fontos lesz, hogy a kerületek nevei betű szerint egyezzenek.

A *Table/Maintenance/Table Structure…* lehetőségre kattintva kiválaszthatjuk, hogy melyik táblázatot kívánjuk szerkeszteni. Ennek kiválasztása után felugrik egy ablak.



4. ábra: *A táblázatunk módosítása*

Az *Add Field* gombra kattintva hozzáadhatunk egy újabb oszlopot, az információknál pedig megadhatjuk a kívánt nevet és az adat típusát. Ez többféle lehet és fontos jól beállítani az adatok helyes kezelése érdekében. Mivel először a kerületek nevét szeretnénk megadni, a *Character* típust választjuk, ebben az esetben megadható egy maximális karakterszám, amit érdemes olyan értékre állítani, aminél hosszabb nevünk feltehetően nem lesz.

Számunkra a továbbiakban még fontos típus az *Integer*, amelyet egész számoknál, illetve a *Float*, amelyet pedig nem egész számok esetén használunk.

Fontos bejelölnünk minden adat esetében, hogy indexelt legyen, mert az indexelt mezők alapján tudunk majd geokódolni.

Ami még esetleg a számunkra lényeges lehet, hogy változtatni tudjuk az oszlopok sorrendjét a táblázatunkban az *Up* és *Down* gombok segítségével.

A beállítások jóváhagyása után, a *Browser* ablakot újra megnyitva megjelenik a kibővült táblázatunk, melybe már beírhatjuk a kerületek neveit latin betűs átírással.

BgRegion_region_stat Browser

Region	Ime	name_latin
SOF	СОФИЯ - ГРАД	
VID	ВИДИН	
VRC	ВРАЦА	
PVN	ПЛЕВЕН	
RSE	РУСЕ	
VTR	ВЕЛИКО ТЪРНОВО	
BLG	БЛАГОЕВГРАД	
BGS	БУРГАС	
GAB	ГАБРОВО	
KRZ	КЪРДЖАЛИ	
KNL	КЮСТЕНДИЛ	
LOV	ЛОВЕЧ	
MON	МОНТАНА	
PAZ	ПАЗАРДЖИК	
PER	ПЕРНИК	
PDV	ПЛОВДИВ	
RAZ	РАЗГРАД	
SLV	СЛИВЕН	
SML	СМОЛЯН	
SFO	СОФИЯ	
SZR	СТАРА ЗАГОРА	

BgRegion_region_stat Browser

Region	Ime	name_latin
SOF	СОФИЯ - ГРАД	Sofia cap.
VID	ВИДИН	Vidin
VRC	ВРАЦА	Vratsa
PVN	ПЛЕВЕН	Pleven
RSE	РУСЕ	Ruse
VTR	ВЕЛИКО ТЪРНОВО	Veliko Tarnovo
BLG	БЛАГОЕВГРАД	Blagoevgrad
BGS	БУРГАС	Burgas
GAB	ГАБРОВО	Gabrovo
KRZ	КЪРДЖАЛИ	Kardzhali
KNL	КЮСТЕНДИЛ	Kyustendil
LOV	ЛОВЕЧ	Lovech
MON	МОНТАНА	Montana
PAZ	ПАЗАРДЖИК	Pazardzhik
PER	ПЕРНИК	Pernik
PDV	ПЛОВДИВ	Plovdiv
RAZ	РАЗГРАД	Razgrad
SLV	СЛИВЕН	Sliven
SML	СМОЛЯН	Smolyan
SFO	СОФИЯ	Sofia
SZR	СТАРА ЗАГОРА	Stara Zagora

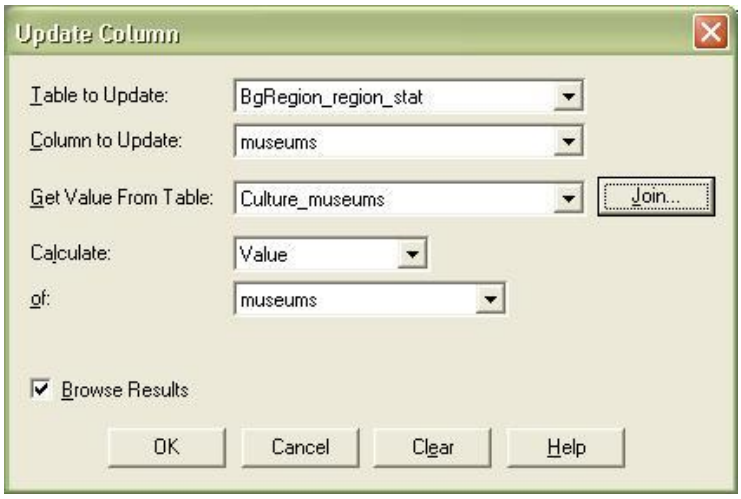
5. ábra: *Kibővített táblázatunk üres, illetve adatokkal feltöltött állapotban*

Ezek után a dolgunk már csak a táblázat feltöltése adatokkal. Ezt tehetnénk az előzőekben leírt módon, vagyis hogy hozzáadunk a táblázathoz egy új, üres oszlopot és abba beírogatjuk az adatokat egyenként. Ez a módszer azon felül, hogy időigényes, még magában hordozza az adatok elírásának, illetve felcserélésének lehetőségét is. Emiatt egy másik módszert alkalmazunk.

Az .xls formátumú fájlt, ami tartalmazza az adatokat, megnyitjuk a MapInfóban. Ugyanúgy, mint a Shapefile esetén, ezt is el kell menteni .tab formátumba, vagyis egyből felugrik egy ablak, melyben megadhatjuk, hogy az .xls fájl mely részét kívánjuk használni. Mivel a letöltött adatsorok több évre vonatkozó adatokat is tartalmaznak, ki kell választatunk a számunkra szükségeset. Ezen felül bejelölhetjük, hogy a kiválasztott állomány első sora adja-e az oszlopaink megnevezését. A beállításokat jóváhagyva a következő ablakban a táblázatunk mezőire (oszlopaira) vonatkozó beállításokat végezhetünk. Megadhatjuk az oszlop nevét, illetve ami igazán lényeges, az adatok típusát.

Fontos, hogy mielőtt megnyitjuk az .xls állományt, töröljük belőle az egyesített *(vagy összevont?)* cellákat, mert a MapInfo azokat nem tudja kezelni. Végül az *OK* gombra kattintva egy *Browser* ablakbanban megjelenik a táblázatunk.

Ebből a táblázatból már igen egyszerűen feltölthetjük adatokkal az előzőekben létrehozott, a kerületekre vonatkozó adatokat tartalmazó táblázatunkat. Először egy üres oszlopot kell hozzáadni a táblázathoz, figyelve arra, hogy az adattípust helyesen adjuk meg. Ezután a *Table/Update Column…* parancsra kattintva megjelenik egy párbeszédpanel.



6. ábra: *Táblázatunk oszlopának frissítése*

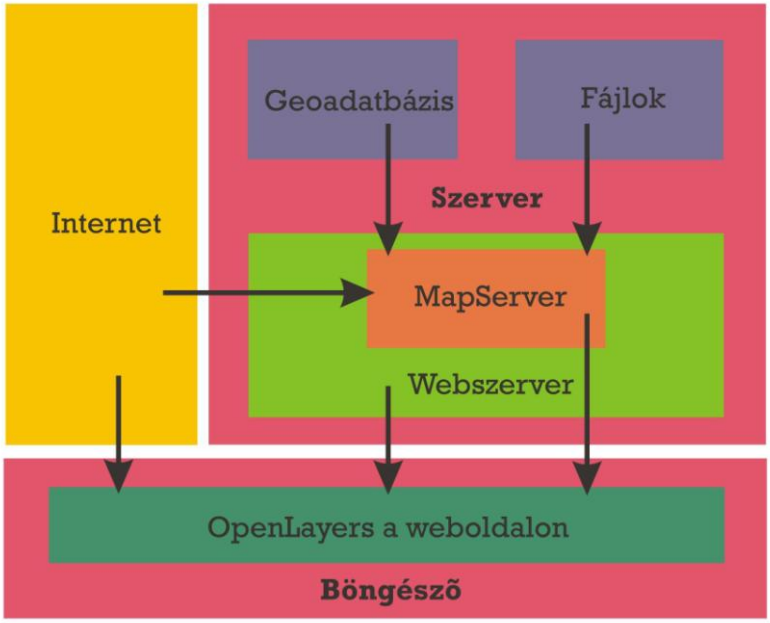
Ebben kell beállítanunk, hogy melyik táblázat (*Table to Update*) melyik oszlopát (*Column to Update*) szeretnénk frissíteni, és hogy melyik táblázat alapján (*Get Value From Table*). A *Join…* gombra kattintva megadhatjuk, hogy mi szerint történjen az adatok párosítása. Itt van jelentősége annak, hogy van már egy-egy oszlopunk a kerületek angol átírású neveivel mindkét táblázatban. Végül beállíthatjuk, hogy a frissítéshez használt táblázat mely adatai (oszlopa) kerüljenek be a táblázatunkba.

Ha mindent beállítottunk a megfelelő módon és az *OK* gombra kattintunk, láthatjuk a kibővült táblázatot, melyet a fent leírt lépések alapján bővítünk tovább, tetszőleges számú oszlopot hozzáadva.

IV. A MapServer

A MapServer egy olyan ingyenes – vagyis nyílt forráskódú - program, melyet eredetileg a University of Minnesota fejlesztett a NASA számára. Ma már egyre szélesebb körben használják, hiszen bárki számára elérhető. Segítségével a weben különböző térképi tulajdonságokkal rendelkező állományokat tudunk megjeleníteni. Nagy előnye, hogy mind raszteres, mind vektoros adatok széles skálájával dolgozhatunk; az általunk megfogalmazott lekérdezések nyomán térképet állít elő.

A MapServer alapvetően egy CGI program, mely a webszerverünkön található. A CGI (Common Gateway Interface) egy olyan szabvány, amely külső programokat, alkalmazásokat információs szerverekkel (például webszerver) köt össze. Ha egy lekérdezést küldünk a webszerverünknek, akkor az továbbítja azt a MapServernek; ekkor igazából egy adott fájl helyére hivatkozunk. Ezt az információt és az úgynevezett mapfile-t felhasználva készít el a MapServer például egy térképet. A lekérdezés nyomán létrejött térkép megjelenítése az OpenLayers segítségével történik, mely egy szintén nyílt forráskódú JavaScript függvénykönyvtár. Ennek használatáról a következő fejezetben lesz szó.



7. ábra: *A MapServer működése*

IV.1. A mapfile

A MapServer működéséhez elengedhetetlen a mapfile, mely .map kiterjesztésű szöveges állomány. Információkat tartalmaz a térképünkre vonatkozóan, hogy hol találhatók az adatok; ebben kerülnek meghatározásra a térképünk rétegei, a vetülete és a használt szimbólumok, stílusok.

A mapfile különböző objektumokból épül fel és minden objektumhoz számos paraméter adható meg. Az objektumok szekciók, vagyis az objektum nevével kezdődnek és minden esetben egy `END` taggal végződnek. Az objektumok hierarchikus rendben követik egymást; a legalapvetőbb objektum a `MAP`. A mapfile minden esetben ezzel kezdődik és a hozzá tartozó `END` taggal végződik.

Hogy hogyan épül fel egy mapfile, egy egyszerű példa alapján át tudjuk tekinteni.

```
MAP
 NAME "pelda"
 STATUS ON
 SIZE 800 600
 IMAGETYPE png
 SHAPEPATH "/home/bscfold2007/vaaoaft/mapserver_data/bg"
 EXTENT 22 41 30 45
 PROJECTION
  "init=epsg:4326"
 END
 FONTSET "fonts.list"
 WEB
  METADATA
    "wms_title" "pelda terkep"
    "wms_srs" "EPSG:4326 EPSG:3857"
  END
 END

 LAYER
  NAME "regio"
  METADATA
    "wms_title" "Regio"
  END
  TYPE polygon
  STATUS DEFAULT
  DATA "BgRegion_region.shp"
  PROJECTION
   "init=epsg:4326"
  END
  CLASS
   NAME "default"
   STYLE
    OUTLINECOLOR 180 50 100
    COLOR 255 230 240
   END
  END
 END
END
```

Az első paraméter a NAME, vagyis a név, melyet idézőjelbe írva kell megadnunk.

A STATUS lehet ON, OFF, vagy DEFAULT; akkor választjuk az OFF állapotot, ha nem szeretnénk, hogy a térképünk látszódjon, ellenkező esetben – és ez a gyakoribb – ON vagy DEFAULT értéket adunk neki. DEFAULT érték esetén a térképünk mindig látható, míg ha ON-ra állítjuk, akkor eldönthető, hogy látszódjon-e vagy sem.

A SIZE paraméter esetében két számot kell megadnunk, ezek a megjelenítendő térkép méretét - az első szám a szélességét, a második a magasságát - adja meg pixelben.

Az IMAGETYPE paraméter segítségével adhatjuk meg, hogy milyen formátumú legyen a megjelenített kép. Azon kívül, hogy vektoros, vagy raszteres formátumot szeretnénk-e számos formátum közül választhatunk.

A SHAPEPATH a térkép készítéséhez felhasznált adatfájl elérési útvonalát határozza meg, itt megadhatunk abszolút elérési útvonalat, vagy a mapfile helyéhez viszonyított elérési utat.

A térkép térbeli – földrajzi – kiterjedését az EXTENT paraméter mellett álló négy számmal adhatjuk meg, ezek értéke függ a vetülettől, melyet a PROJECTION paraméterrel határozunk meg. A megjelenítendő térképünk vetületét legegyszerűbben egy EPSG azonosító segítségével adhatjuk meg, amely egy adott vetület minden paraméterét tartalmazza. Ez az azonosító a MapServer által is használt, **Proj.4** vetületi átszámító függvénykönyvtárban eltárolt térképi vetületekre hivatkozik. Ha minden általunk használt adat ugyanabban a vetületben van, nem szükséges megadnunk ezt a paramétert.

A FONTSET segítségével határozzuk meg a használni kívánt betűtípust, melyből több is megadható. Itt valójában egy listára hivatkozunk – abszolút vagy relatív elérési utat megadva – , mely tartalmazza ezen betűtípusokat. A betűtípusokat tartalmazó .ttf állományokat ott kell tárolnunk, ahol maga a lista is található.

A WEB objektum paramétereit akkor kell meghatároznunk, ha a MapServert WMS szolgáltatásként használjuk. A WMS a Web Map Service rövidítése; ez egy az Open Geospatial Consortium (OGC) által meghatározott szabvány különböző adatbázisokból létrehozott térképek megjelenítéséhez az interneten. A wms_title a térkép nevét határozza meg, a wms_srs segítségével pedig azt adhatjuk meg, hogy a megjelenítendő térképünk milyen vetületben lesz (lehet).

Most lássuk, hogyan is néz ki a térképnek egy rétege, vagyis a LAYER.

Célszerűen először itt is a nevet adjuk meg, mert később, amikor már több rétegünk lesz, átláthatóbb, ha mindenhol a név az első paraméter. A METADATA objektumban itt is a WMS szolgáltatásként való alkalmazás paramétereit adhatjuk meg.

A TYPE paraméter határozza meg az adatok ábrázolási formáját; többféle értéket is felvehet:

- point (pont), line (vonal) vagy polygon (sokszög): ezek esetében fontos, hogy a forrásul szolgáló Shapefile-lal megegyező típust adjunk meg, mert míg egy sokszög megjeleníthető pontként, ugyanez fordítva nem működik.

- annotation (megírás): segítségével a különböző objektumok neve megjeleníthető anélkül, hogy maga az objektum megjelenítésre kerülne.

- chart (diagram): tematikus térkép készítése esetén diagramokat tudunk definiálni

- circle (kör): a legkisebb befoglaló négyszög méretét megadva kört határozhatunk meg.

- query (lekérdezés): ha a réteg típusának ezt választjuk, akkor a réteg lekérdezhető lesz, de nem kerül megjelenítésre.

- raster (raszter): az adatok egy raszteres kép formájában jelennek meg.

A STATUS paraméter itt is ugyanúgy megadható; a PROJECTION paraméter azonban itt azt mondja meg, hogy az adataink milyen vetületben vannak.

A DATA paraméter a MAP objektum SHAPEPATH paraméteréhez viszonyítva megadja a felhasználandó adatokat tároló állományunk nevét.

Egy rétegen belül különböző osztályokat, CLASS-okat hozhatunk létre. Ennek segítségével tetszőleges, általunk meghatározott szempontok szerint csoportosíthatjuk és jeleníthetjük meg az adatokat. A NAME paraméterrel nevet adunk a csoportnak. A STYLE paraméterei pedig a megjelenítésre vonatkoznak; például egy sokszög (polygon) esetében többek között megadható a kitöltés és a kontúrvonal színe is.

A LAYER objektum is END taggal végződik.

8. ábra: *A mapfile definiálta térkép*

A fentiekben egy igen egyszerű mapfile felépítését ismertük meg. Az objektumok – adott térképi tartalom megjelenítéséhez kapcsolódó – részletesebb bemutatása a későbbiekben következik.

IV.2. Tematikus térképek definiálása a mapfile-ban

Azt már láttuk, hogy egy térkép – tematikus térkép esetében a háttértérkép – elemeit hogyan tudjuk megjeleníteni a MapServer segítségével.

Most tematikus térképeket szeretnénk létre hozni. A tematikus térképek készítéséhez adatokra van szükségünk, melyeket a MapServer számára elérhető helyen és formátumban kell tárolnunk. A dolgozat elején tárgyalt módon összekapcsoltuk a különböző objektumokat a rájuk vonatkozó adatokkal egy .tab kiterjesztésű állományban, így azokat megfelelő helyen tárolva fel tudjuk használni különböző tematikus térképek előállításához.

IV.2.1. A kartogrammódszer

A legegyszerűbb módja a tematikus adatok ábrázolásának, hogy a különböző közigazgatási egységekre vonatkozó mennyiségi adatokat a színkitöltés milyenségével jelenítjük meg.

Vegyünk például egy olyan tematikus térképet, amely a Bulgária kerületeiben található múzeumok számát jeleníti meg. A térkép megjelenítéséhez szükséges mapfile-nak a tematikus adatokat megjelenítő rétege (LAYER) a következőképpen néz ki:

```
LAYER
  NAME "regions_museums"
  METADATA
    "wms_title" "Regions_museums"
  END
  TYPE polygon
  CONNECTIONTYPE OGR
  CONNECTION "BgRegion_region_stat.TAB"
  PROJECTION
   "init=epsg:4326"
  END
  STATUS ON
  CLASS
   NAME "3 db alatt"
   EXPRESSION ([museums]<=2)
   STYLE
    OUTLINECOLOR 120 120 120
    COLOR 220 207 255
   END
  END
  CLASS
   NAME "3-5 db"
   EXPRESSION ([museums]<=5 and [museums]>2)
   STYLE
    OUTLINECOLOR 120 120 120
    COLOR 187 161 255
   END
  END
  CLASS
   NAME "6-10 db"
   EXPRESSION ([museums]<=10 and [museums]>5)
   STYLE
    OUTLINECOLOR 120 120 120
    COLOR 153 113 255
   END
  END
  CLASS
   NAME "11-20 db"
   EXPRESSION ([museums]<=20 and [museums]>10)
   STYLE
    OUTLINECOLOR 120 120 120
    COLOR 120 67 255
   END
  END
  CLASS
   NAME "20 db felett"
   EXPRESSION (21<[museums])
   STYLE
    OUTLINECOLOR 120 120 120
    COLOR 65 0 229
   END
  END
 END
```

A LAYER két új paramétere a CONNECTION és a CONNECTIONTYPE. Az első megadja, hogy mely fájlhoz kapcsolódjon a MapServer az adatok elérése érdekében, a második a kapcsolat típusát határozza meg. Az OGR egy nyílt forráskódú könyvtár, amelynek segítségével a Shapefile-on kívüli számos vektoros adat megjelenítése lehetővé válik. Mivel esetünkben az adatok egy .tab kiterjesztésű állományban vannak tárolva, ezt választjuk.

Az adatokat osztályozzuk; a különböző osztályokat a CLASS-ok segítségével különítjük el. A NAME paraméter segítségével nevet adhatunk az adott osztálynak.

Az EXPRESSION paraméter azt mondja meg, hogy a .tab fájl mely adatát (oszlopát) kell figyelembe venni illetve, hogy annak mely értékét. Például a „6-10 db" nevű osztályba tartoznak azok a kerületek, ahol a [museums] értéke kisebb vagy egyenlő tízzel és nagyobb ötnél.

A STYLE objektumon belül pedig az adott osztály térképen történő megjelenésének milyenségét határozhatjuk meg.

IV.2.2. A diagrammódszer

A mapfile-ban definiálni tudunk diagramokat is. Vegyük példának a kerületek népességének nemek szerinti megoszlását. Az oszlopdiagramokat tartalmazó réteg meghatározása a mapfile-ban a következő:

```
LAYER
  NAME "regions_population_malefemale"
  METADATA
    "wms_title" "Regions_population_malefemale"
  END
  TYPE chart
  CONNECTIONTYPE OGR
  CONNECTION "BgRegion_region_stat.TAB"
  PROCESSING "CHART_TYPE=BAR"
  PROCESSING "CHART_SIZE=20 80"
  PROCESSING "CHART_BAR_MAXVAL=800000"
  STATUS ON
  PROJECTION
   "init=epsg:4326"
  END
  CLASS
   NAME "férfiak"
   STYLE
    SIZE [population_male]
    COLOR 71 130 169
   END
  END
  CLASS
```

```
    NAME "nők"
    STYLE
     SIZE [population_female]
     COLOR 219 124 51
    END
   END
  END
```

A réteg típusa `chart`, vagyis diagram.

A `PROCESSING` paraméter segítségével tudjuk meghatározni a megrajzolandó diagramok típusát és méretét. A `"CHART_TYPE=BAR"` jelenti azt, hogy oszlopdiagramot készítünk. A méretét a `CHART_SIZE` paraméter határozza meg, ahol az első szám a diagram szélessége, a második pedig a legmagasabb értékhez tartozó magassága; illetve a méretnél van szerepe a `CHART_BAR_MAXVAL` paraméternek is, ugyanis ennek segítségével megadhatjuk a legnagyobb számértékű adatot, amit még ábrázolni kívánunk. Ha ennél nagyobb szám is szerepel az adataink között, akkor az ahhoz tartozó oszlopdiagram mérete nem fogja meghaladni az általunk meghatározott legnagyobb adathoz tartozó diagram méretét. Hasonló a szerepe a `CHART_BAR_MINVAL` paraméternek is, amelynek segítségével egy alsó határt szabhatunk az ábrázolni kívánt adatoknak. Az ez alá eső számértékek diagrammérete is az általunk meghatározott legkisebb értékhez tartozó diagram méretét fogják felvenni.

A `CLASS`-ok meghatározásánál a `STYLE` objektumon belül, a `SIZE` paraméternél adjuk meg a szín mellett, hogy mely adat alapján számítsa a MapServer a diagram méretét.

Nem csak oszlopdiagramot tudunk készíteni, hanem kördiagramot is. Vegyünk itt is egy példát; a kerületenként élve születettek nemek szerinti arányát ábrázoló tematikus réteget. A diagramokat tartalmazó réteg a következő:

```
LAYER
  NAME "regions_livebirth_boysgirls"
  METADATA
    "wms_title" "Regions_livebirth_boysgirls"
  END
  TYPE chart
  CONNECTIONTYPE OGR
  CONNECTION "BgRegion_region_stat.TAB"
  PROCESSING "CHART_TYPE=PIE"
  PROCESSING "CHART_SIZE_RANGE= live_birth_total 30 70 829 75513"
  STATUS ON
  PROJECTION
   "init=epsg:4326"
  END
  CLASS
   NAME "fiúk"
   STYLE
    SIZE [live_birth_boys]
    COLOR 60 83 111
```

```
  END
 END
 CLASS
  NAME "lányok"
  STYLE
   SIZE [live_birth_girls]
   COLOR 180 116 167
  END
 END
END
```

A `"CHART_TYPE=PIE"` határozza meg, hogy kördiagramot készítünk, de ez a paraméter elhagyható, mert a kördiagram az alapértelmezett diagramtípus. Ha azt szeretnénk, hogy minden diagramunk azonos méretű legyen, akkor elég csak megadnunk a kívánt átmérőt a `"CHART_SIZE=`*a*`"` paraméter segítségével, ahol *a* az átmérő mérete pixelben megadva. Azonban a diagram méretezhető egy adott adat nagyságához is. Nézzük meg a példánkat: `"CHART_SIZE_RANGE= live_birth_total 30 70 829 75513"`, ahol a diagram méretét az összes élve született gyermek számához igazítjuk. A következő két szám segítségével határozhatjuk meg a legkisebb, illetve a legnagyobb méretét a diagramunknak pixelben megadva, az utolsó két szám pedig azt mondja meg, hogy mi a legkisebb számadat, amit a legkisebb mérethez, illetve mi a legnagyobb számadat, amit a legnagyobb mérethez rendelünk hozzá.

IV.3. Bulgária térképének rétegeit tartalmazó mapfile létrehozása

Az előző fejezetekben megismertük, hogy hogyan épül fel egy mapfile, illetve, hogy abban hogyan tudunk tematikus térképet definiálni. Most ezen ismeretek birtokában készítjük el a mapfile-unkat, melynek segítségével Bulgáriáról készült térképeket, tematikus térképeket jeleníthetünk meg a weben.

A különböző tartalmakat külön rétegeken jelenítjük meg, melyeknek száma igen nagy lehet. Célszerű ezért azokat valamilyen szempont alapján sorba rendezni a mapfile-on belül, illetve olyan névvel látni el őket, amelyek egyértelműen utalnak a rétegen megjelenített tartalomra.

Mivel a MapServert WMS szolgáltatásként használjuk, fontos, hogy itt is és minden további rétegben szerepeljen egy a nevet, a vetületet és a metaadatokat tartalmazó elem; vagyis egy `NAME`, egy `PROJECTION` és egy `METADATA` objektum.

Szintén minden réteg esetében a STATUS paramétert ON értékre állítjuk, így mindegyik rétegnél eldönthető lesz, hogy látszódjon-e vagy sem.

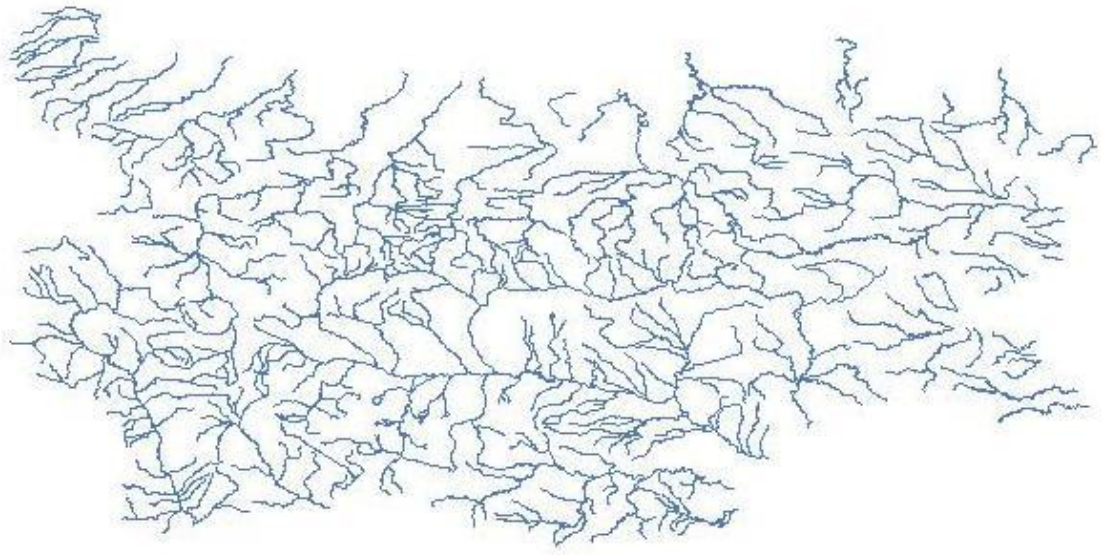
IV.3.1. Tematikus adatokat nem tartalmazó rétegek létrehozása

Először néhány olyan réteget definiálunk, mely egy tematikus tartalommal nem rendelkező térkép létrehozására alkalmas.

Az első réteg Bulgária határát tartalmazza. A réteg típusa polygon, vagyis sokszög. Ez a réteg csak egyetlen osztályt tartalmaz – és igazából csak egyetlen elemet: az országhatárt. Az osztály STYLE objektumán belül megadjuk ennek az elemnek a stílusát; a színét az OUTLINECOLOR és a vastagságát a WIDTH paraméterek segítségével. A mapfile-ban a színeket az RGB színmodell alapján határozzuk meg. Az egyes csatornák intenzitásértékét egy 0 és egy 255 közötti, egész számmal adjuk meg. A vonal – vagy kontúrvonal – vastagságát pixelben adjuk meg. Ha nem határozunk meg más értéket, akkor alapesetben a vonal vastagsága egy pixel. A határvonal öt pixel szélességű.

A határvonalat tartalmazó réteghez nagyon hasonlók a kerületeket, a tavakat és a folyókat tartalmazó rétegek; ezek esetében is csak egy-egy osztályt határozunk meg, mely az összes ábrázolandó objektumot magába foglalja.

A folyókat két rétegre különítjük el; az egyik sokszög típusú, mert a rendelkezésünkre álló Shapefile a Dunát sokszögként tartalmazza, az összes többi folyót pedig vonalként, így azok egy vonal típusú rétegre kerülnek. Ebben az esetben célszerű létrehoznunk egy rétegcsoportot, mely magába foglalja a két folyókat tartalmazó réteget. Ezt könnyen megtehetjük a GROUP paraméter segítségével – ahol a csoport nevét idézőjelbe téve kell megírnunk –, melyet azon rétegek esetében kell megadnunk, amelyeket csoportosítani szeretnénk. Így ha adott esetben a folyókat kívánjunk megjeleníteni, akkor a rétegekre külön-külön történő hivatkozás helyett elég a rétegcsoportra hivatkoznunk.



9. ábra: *A folyókat ábrázoló vonal típusú réteg*

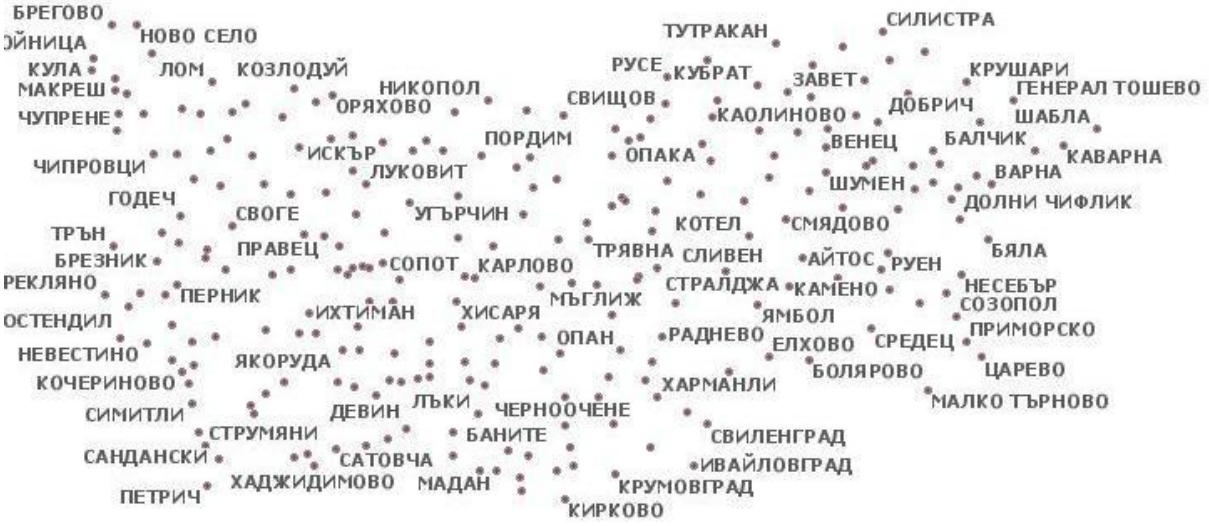
A színeket a magyar kartográfiai hagyományoknak megfelelően választva az országhatár pirosas színt kap, míg a vízrajz kék színű. A régiók kontúrvonalának színe megegyezik az országhatáréval. A folyók és a tavak kontúrvonala is azonos színű, a tavak kitöltése pedig egy halványabb árnyalatú kék.

Még egy réteg tartozik ebbe a kategóriába; ez egy pont `(point)` típusú réteg, amely a városokat tartalmazza. Ezen réteg meghatározó objektuma a `LABELITEM`, mely megszabja, hogy az adatbázisunk mely adatából készüljön a megírás. Egyetlen osztályt hozunk létre; minden név abba tartozik. Az osztály objektumon belül tudjuk a pontok jelölésének milyenségét meghatározni.

A `SYMBOL` paraméter egy már általunk a `MAP` objektumon belül definiált szimbólumra hivatkozik, amely most a városok esetében kör. Meghatározzuk a méretét a `SIZE` és a színét a `COLOR` illetve `OUTLINECOLOR` paraméterekkel.

Feliratot is adunk hozzá a `LABEL` objektum által. Típusa `truetype`; a `FONT` paraméter `trebuchet_bold`. Az azonosítót, mellyel a betűtípusra hivatkozunk, a betűtípusok listája szintén tartalmazza. A `SIZE` a betűméretet, a `COLOR` a betű színét, az `OUTLINECOLOR` pedig a betű kontúrjának színét határozza meg. A betűszínt szürkének választjuk, ezért érdemes egy fehér kontúrt adni neki, hogy minden szín előtt jól olvasható legyen. A `POSITION` paraméter többféle értéket felvehet; ez határozza meg a megírás elhelyezkedését a jelölt objektumhoz – sokszög esetében annak középpontjához – képest. A városok esetében az `auto` értéket állítjuk be, mert ekkor a MapServer arra törekszik, hogy a megírásokat úgy helyezze el, hogy a lehető

legtöbb megjelenjen a térképen. Mivel a réteg cirill neveket tartalmaz, elengedhetetlen az `ENCODING` paraméter megadása, hogy a cirill betűket meg lehessen jeleníteni. A Windows-1251-es karakterkódolást használjuk.



10. ábra: *A városokat ábrázoló réteg*

Minden eddigi réteg esetében a `DATA` paraméter a megfelelő adatokat tartalmazó Shapefile-ra hivatkozik.

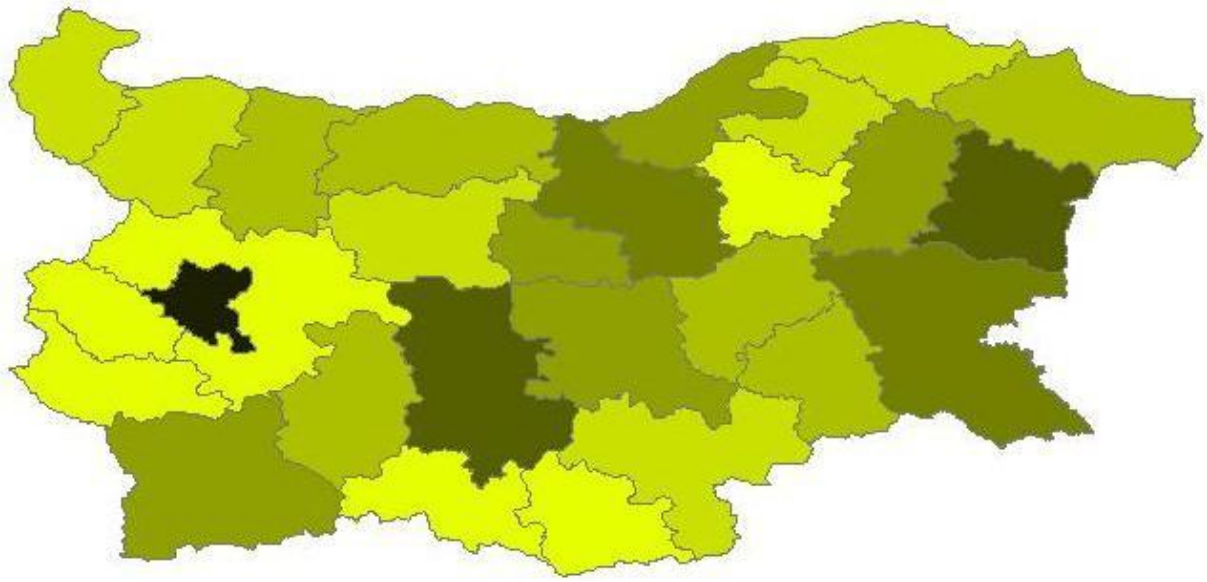
IV.3.2. Tematikus adatokat tartalmazó rétegek létrehozása

A bolgár Nemzeti Statisztika Intézet honlapján hatalmas mennyiségű adat lelhető fel. A MapServer segítségével készíthető tematikus térképek bemutatásához két témakört választunk és mutatunk be érintőlegesen.

Az első néhány rétegen kultúrával kapcsolatos tematikák jelennek meg; ezek mennyiségi adatok, melyek az egyes kerületekre vonatkoznak. Ábrázolásukra a kartogrammódszert alkalmazzuk, így ezek a rétegek sokszög típusúak. A négy réteg a következő:

- Megjelent újságok száma
- Kiadott könyvek száma
- Múzeumok száma
- Múzeumi látogatások száma

Az adatok osztályozása a már ismertetett módon, CLASS-ok segítségével történik. Az, hogy hány osztályt hozunk létre, illetve hogy milyen kategóriahatárokat választunk mindig az adott tematikától függ. Fontos, hogy ezeket a határokat az adatokhoz, azok eloszlásához igazodva válasszuk meg, így térképünk szemléletes lesz. Ezt figyelembe véve a megjelent újságok és múzeumok számának ábrázolásánál öt-öt, a múzeumi látogatások számának ábrázolásánál hat, a kiadott könyvek számának ábrázolása esetében pedig hét osztályt különítünk el.

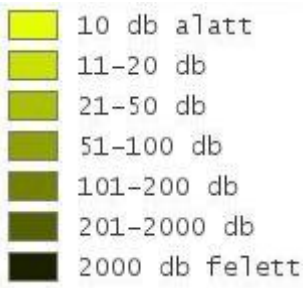


11. ábra: *A kerületenként kiadott könyvek számát ábrázoló réteg*

A mapfile-ban minden osztálynál meghatározunk egy név paraméter is. A MapServer a jelmagyarázatot az osztályokból készíti, tehát azok nevét úgy célszerű megadnunk, ahogyan a jelmagyarázatban szeretnénk látni. Ha a név paramétert elhagyjuk, akkor az adott osztály nem fog megjelenni a jelmagyarázatban.

Az egyes osztályok STYLE objektumán belül a kontúrvonal illetve a kitöltés színét határozzuk meg. A kitöltés színét úgy választjuk meg, hogy szemléltessék a kerületre vonatkozó adat nagyságát; tehát minél nagyobb a szám, ugyanazon szín annál sötétebb árnyalatát kapja a kerület. A kerületek határa szürke színű.

A jelmagyarázat tulajdonságait a LEGEND objektumon belül tudjuk meghatározni. A KEYSIZE paraméter megadja a jelmagyarázatban megjelenő felületek méretét pixelben. A LABEL objektumban az eddigiekhez hasonló módon a betűtípusra, a betű színére, méretére és elhelyezkedésére vonatkozó adatokat adunk meg. A STATUS paraméter értéke ON.



12. ábra: *A kerületenként kiadott könyvek számát ábrázoló réteghez tartozó jelmagyarázat*

A többi rétegen a lakosságra vonatkozó tematikus adatokat ábrázolunk, szintén kerületenként. Itt a kartogrammódszer mellett már a diagrammódszert is alkalmazzuk. A sokszög típusú rétegek tehát a következők:

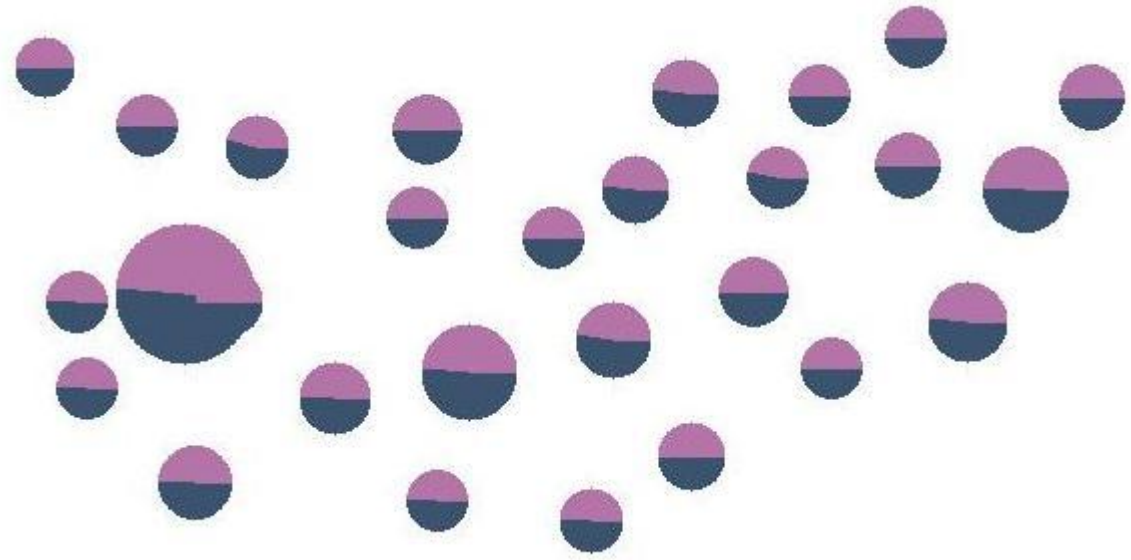
- Termékenységi ráta
- Élve születettek száma
- Halálozási ráta
- Népesség
- Megkötött házasságok száma
- Válások száma

Az adatok osztályozása az előzőekben leírtakhoz hasonlóan itt is mindig a konkrét adatsortól függ. A termékenységi rátát, a népesség számát, és a megkötött házasságok számát ábrázoló rétegeken hét, a többi rétegen pedig hat osztályt különítünk el.

A tematikus adatokat diagrammódszerrel megjelenítő rétegek típusának a `chart`-ot, vagyis diagramot állítjuk be. Három ilyen réteget hozunk létre:

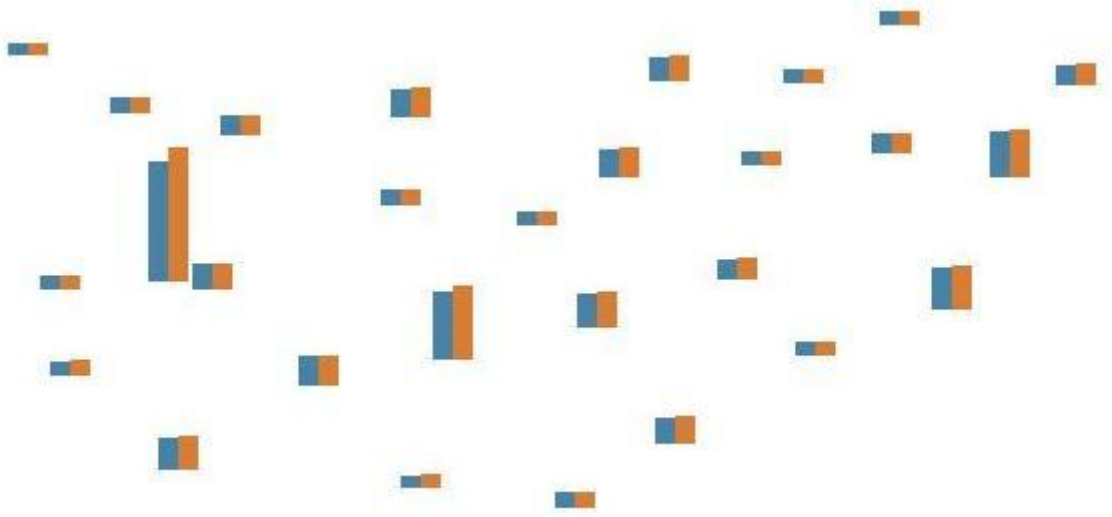
- Élve születettek nemek szerinti megoszlása
- Halálozási ráta nemek szerinti megoszlása
- Népesség nemek szerinti megoszlása

Az első két rétegen kördiagramot hozunk létre. Mindkét esetben a diagram méretével is érzékeltetni szeretnénk az adott kerületre vonatkozó élve születettek számát, illetve a halálozási ráta nagyságát. Emiatt a `CHART_SIZE` paraméter helyett a `CHART_SIZE_RANGE` paramétert használjuk, ahol mi határozhatjuk meg a diagram legkisebb, illetve legnagyobb méretét.



13. ábra: *Az élve születettek nemek szerinti megoszlását ábrázoló réteg*

A népesség nemek szerinti megoszlását oszlopdiagramon ábrázoljuk, tehát `CHART_TYPE=BAR`. Itt a `CHART_SIZE` paraméter mellett a `CHART_SIZE_RANGE` helyett a `CHART_BAR_MAXVAL` paramétert használjuk. A `CHART_SIZE` segítségével meghatározzuk az oszlopdiagram szélességét és a legnagyobb értékhez tartozó magasságát, a `CHART_BAR_MAXVAL`-lal pedig meghatározzuk, hogy mekkora ez a legnagyobb érték.



14. ábra: *A népesség nemek szerinti megoszlását ábrázoló réteg*

A fent felsoroltak mellett létrehozunk továbbá `annotation` típusú rétegeket, vagyis olyan rétegeket, amelyek csak megírásokat tartalmaznak.

Először kettő, a kerületek nevét tartalmazó réteget készítünk; az egyik az eredeti, cirill betűs megírást tartalmazza, a másik pedig a latin betűs átírást. A városokat tartalmazó rétegtől ezek annyiban térnek el, hogy semmilyen objektumot nem jelenítenek meg, csak megírást. Így a `CLASS` elemen belül, csak a feliratokra `(LABEL)` vonatkozó paramétereket kell megadnunk. A cirill nevek megjelenítése miatt itt is meg kell adnunk az `ENCODING` paramétert.

A népesség, a megkötött házasságok száma, és a válások száma estében létrehozunk továbbá olyan rétegeket is, amelyek a kerületek nevei mellett az adott tematika kerületre vonatkozó számértékét is megjeleníti. Itt tehát a felirat `(LABEL)` már kétféle információból tevődik össze. Hogy mi legyen ez a két információ, azt a `CLASS` objektumon belül a `TEXT` paraméter segítségével tudjuk megadni. Nézzük meg ezt a paramétert például a népesség réteg esetében: `TEXT([name_latin];[population])`, ahol a `[name_latin]` az adatbázisunk azon oszlopára hivatkozik, mely a kerületek neveit latin betűs átírással tartalmazza; a `[population]` pedig a kerületek népességét tartalmazó oszlopra.

A `LABEL` objektum itt három új paramétert tartalmaz. Az `ALIGN` paraméter többsoros megírások esetében adható meg; a sorok egymáshoz igazítását szabályozza; értéke lehet, `left`, `center` vagy `right`. Mi most a `center`-t választjuk, vagyis középre igazítunk. A `WRAP` paraméter után idézőjelben megadott karakter, jelen esetben „;" határozza meg, hogy a többsoros megírás esetében hol legyen a sornak vége. A `PRIORITY` paraméter 1 és 10 közötti értéket vehet fel, ahol 10 a legmagasabb. Ha többféle megírás töltődik be a térképünkre, akkor ennek a paraméternek megfelelően fognak sorban megjelenítésre kerülni a feliratok. Mi most ezt 10-re állítjuk, mert a legfontosabb az adatok megjelenése a térképünkön.

A megkötött házasságok és válások számát megjelenítő rétegen a latin betűs átírás helyett a cirill megírás megjelenítését választjuk az adatok mellett.

A tematikus tartalmat hordozó rétegek esetében is csoportosíthatunk rétegeket. Ha például a népesség számának sokszög és megírás típusú rétegei a `GROUP` paraméter segítségével egy csoportot alkotnak, akkor a későbbiekben ha mindkettő réteget meg szeretnénk jeleníteni, akkor csak a csoport nevére kell hivatkoznunk ahelyett, hogy külön-külön tennénk ezt meg a rétegek nevére. Ugyanígy csoportosítjuk a megkötött házasságok számát és a válások számát ábrázoló két-két réteget.

A tematikus rétegek esetében a `CONNECTIONTYPE` és a `CONNECTION` paraméterek megadása biztosítja a szükséges adatokra (.tab állományokra) történő hivatkozást.

IV.4. Néhány megjegyzés a mapfile-lal kapcsolatban

Fontos tudnunk, hogy a MapServer a mapfile-ban meghatározott sorrendben olvassa az adatokat, vagyis az a réteg, amit legelőször definiálunk lesz a térképünk legalsó, amit pedig legutoljára, a legfelső rétege.

A mapfile nem érzékeli a különbséget kis- és nagybetű között. Néhány esetben azonban, amikor a kis- és nagybetűkre érzékeny platformon vagyunk, érdemes odafigyelnünk erre, ha valamilyen file nevére hivatkozunk.

A mapfile-ba megjegyzések szúrhatók, ha elé kettőskeresztet `(#)` írunk.

V. A térkép a weben

Láthattuk, hogy a MapServer hogyan állít elő térképet, különböző térképi rétegeket a rendelkezésünkre álló adatokból.

Ahhoz, hogy ezekből a térképekből egy interaktív térképet tudjunk létrehozni, és azt megjeleníteni egy weboldalon, az OpenLayers-t használjuk.

Az OpenLayers egy nyílt forráskódú JavaScript függvénykönyvtár, amelynek segítségével interaktív térképet tudunk weboldalba ágyazni. Az OpenLayers csak megjelenítésre szolgál, de segítségével különböző forrásokból származó adatokat tudunk egy térkép előállításához felhasználni. Ezek az adatforrások sokfélék lehetnek; raszteres vagy vektoros adatok, vagy például WMS szerver szolgáltatta adatok.

V.1. Az OpenLayers használata

Ahhoz, hogy az OpenLayers JavaScript könyvtárának egész eszközkészletét használni tudjunk, a weboldalunk – melyen a térképet szeretnénk megjeleníteni – html kódjának fejrészében (<head>) hivatkoznunk kell rá, a <script> elemen belül. Ehhez tartozik egy elem a szövegtörzsben (<body>), ez adja meg a térkép helyét, illetve itt tudjuk meghatározni, hogy mekkora legyen a térkép által elfoglalt terület pixelben:

```
<html>
 <head>
  <title>OpenLayers - Bulgária</title>
  <script src="http://openlayers.org/api/OpenLayers.js"></script>
 </head>
 <body>
  <div style="width:600px; height:400px" id="terkep_helye"></div></td>
  </body>
</html>
```

Ez még csak maga a hivatkozás, most létre kell hoznunk magát a térképet. Két változót definiálunk, az elsőt a térkép objektum tárolására:

```
var map;
map = new OpenLayers.Map('terkep_helye',
         { projection:"EPSG:3857",
         maxExtent: new OpenLayers.Bounds(2500000,5000000,3100000,5500000),
         maxResolution: 156543.0339,
```

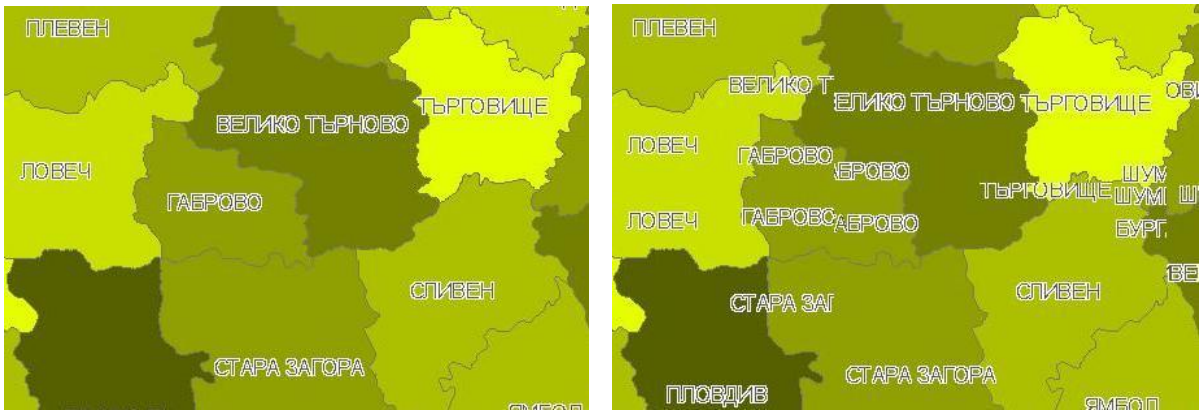
```
        units: "m"
        });
```

A `'terkep_helye'` azonosító szolgál a `<div>` elemmel való összekapcsolásra. Itt több dolgot meghatározhatunk a térképünkkel kapcsolatban. Az egyik a vetület. Az `"EPSG:3857"` a Mercator-féle szögtartó hengervetület. A `maxExtent` paraméter segítségével adjuk meg a térkép legnagyobb kiterjedését, vagyis nem mozdíthatjuk el úgy a térképet, hogy ez a terület teljesen kikerüljön a látómezőből. Ezt a kiterjedést mindig az adott vetületben meghatározható koordináták alapján kell megtennünk. Jelen esetben olyan értékeket választunk, hogy Bulgária területe teljes egészében látható legyen. A `maxResolution` értéke azt adja meg, hogy mi az a legnagyobb felbontás, amikor nem nagyítunk bele a térképbe (ha zoom=0). Ez esetben egy pixel 156543,0339 méternek felel meg. Az egység meghatározása a `units` paraméter segítségével történik.

A másik változót egy WMS réteg tárolására definiáljuk:

```
var wms;
wms = new OpenLayers.Layer.WMS( "Bulgária",
          'http://mercator.elte.hu/cgi-
       bin/mapserv?map=/home/bscfold2007/vaaoaft/mapserver_data/bg3.map',
          {layers: ['border'],isBaseLayer:false,transparent:true},
          {projection:"EPSG:3857",singleTile:true});
```

Ezzel határozzuk meg, hogy a WMS szabvány segítségével szeretnénk az adatokat szolgáltatni. Megadjuk a létrehozandó réteg nevét és a felhasználni kívánt adatok - jelen esetben egy mapfile - elérési útvonalát a szerverünkön, valamint a réteg nevét a mapfile-ban, ami most az országhatárt tartalmazza. Az `isBaseLayer` és a `transparent` paraméterek segítségével tudjuk meghatározni, hogy a rétegünk *base layer*, vagyis olyan réteg, amiből egyszerre csak egy jeleníthető meg, vagy *overlay* legyen. Ha a réteg overlay típusú, akkor más rétegekkel együtt, egyszerre is megjeleníthető. Ahhoz, hogy ilyen réteget kapjunk, az `isBaseLayer` paraméternek `false`, a `transparent` paraméternek pedig `true` értéket kell adnunk. Itt is meghatározzunk a réteg vetületét - mely azonos a térkép vetületével - , a `singleTile` paraméter `true`-ra állításával pedig azt érjük el, hogy az egész réteg egy darabként fog viselkedni, tehát például ha arrébb húzzuk a térképet, akkor az egész réteg újratöltődik. Ennek számunkra azért van jelentősége, mert a térképünkön lesznek feliratok és így, hogy a térkép egy darabból áll, minden név csak egyszer szerepel rajta, míg ha a `false` értéket adnánk ennek a paraméternek, az egyes kerületek nevei többször is megjelennének, attól függően, hány darabra esik a területük.



15. ábra: *A* `singleTile` *paraméter* `true`*, illetve* `false` *értékre állítva*

Ezek után már csak hozzá kell adnunk a réteget a térképünkhöz. Ezt a következő paranccsal tesszük meg:

```
map.addLayer(wms);
```

Egyelőre van egy térképünk, ami csak Bulgária országhatárát tartalmazza. Ahhoz, hogy az oldal betöltésénél legelőször ne csak ezt a szinte teljesen tartalom nélküli térképet lássuk, hozzáadunk egy OSM réteget, vagyis egy réteget, ami az OpenStreetMap-et tartalmazza:

```
map.addLayer(new OpenLayers.Layer.OSM());
```

Ez ad magyarázatot arra, hogy miért a Mercator-féle szögtartó hengervetületet választottuk az előbbiekben, ugyanis az OpenStreetMap csak ebben a vetületben érhető el.

A `map.zoomToExtent` függvény segítségével továbbá megadjuk, hogy az oldal betöltésekor mekkora területet lássunk a térképen. Ez jó, ha nagyjából megegyezik a `maxExtent` paraméternél megadott határokkal.

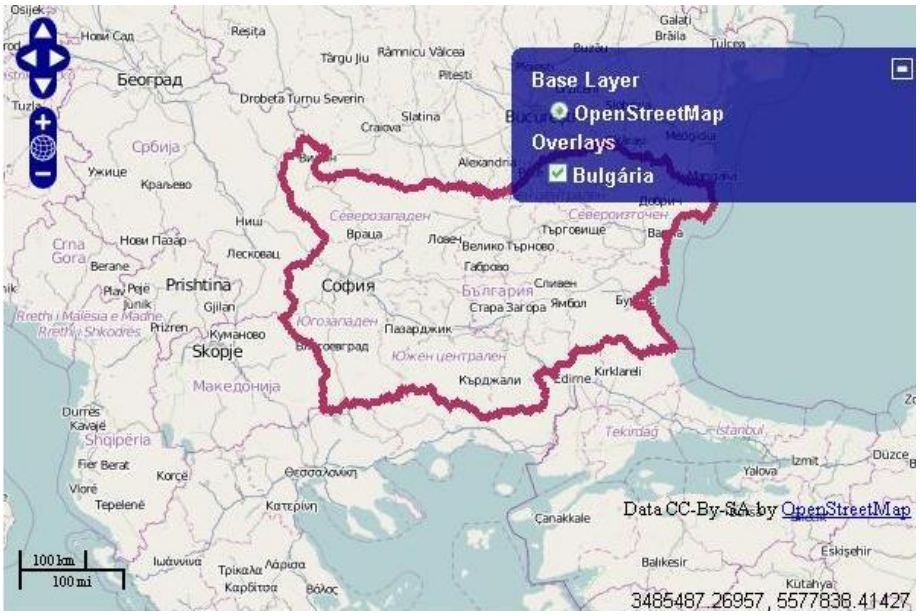
Ekkor a térképünk így néz ki:

16. ábra: *Térképünk megjelenítése OpenLayers segítségével*

Az OpenLayers-nek köszönhetően egyszerűen hozzáadhatunk eszközöket, ún. control objektumokat a térképünkhöz, melyeknek köszönhetően az még informatívabb lesz:

```
map.addControl(new OpenLayers.Control.MousePosition());
map.addControl(new OpenLayers.Control.ScaleLine());
map.addControl(new OpenLayers.Control.LayerSwitcher());
```

Az `OpenLayers.Control.MousePosition` kiírja annak a pontnak a koordinátáit, ahol éppen a kurzor van. Az `OpenLayers.Control.ScaleLine` egy dinamikus mértéklécet definiál, mely metrikus illetve angolszász mértékegységben is kiírja adott egység valóságban megfelelő hosszát. Az `OpenLayers.Control.LayerSwitcher` segítségével pedig rétegek tehetők ki- vagy bekapcsolhatóvá.



17. ábra: *Térképünk az OpenLayers segítségével hozzáadott eszközökkel*

A fenti térkép html kódjának fejrésze a következő:

```
<html>
 <head>
  <meta content="text/html; charset=utf-8" http-equiv="Content-Type" />
  <title>OpenLayers - Bulgária</title>
  <script src="http://openlayers.org/api/OpenLayers.js"></script>
  <script type="text/javascript">
   var map,wms;

   function init()
   {
     map = new OpenLayers.Map('terkep_helye',
         { projection:"EPSG:3857",
         maxExtent: new OpenLayers.Bounds(2500000,5000000,3100000,5500000),
         maxResolution: 156543.0339,
         units: "m"
         });

     wms = new OpenLayers.Layer.WMS( "Bulgária",
               'http://mercator.elte.hu/cgi-
      bin/mapserv?map=/home/bscfold2007/vaaoaft/mapserver_data/bg3.map',
         {layers: ['border'],isBaseLayer:false,transparent:true},
         {projection:"EPSG:3857",singleTile:true});

     map.addLayer(wms);
     map.addLayer(new OpenLayers.Layer.OSM());
     map.addControl(new OpenLayers.Control.MousePosition());
     map.addControl(new OpenLayers.Control.ScaleLine());
     map.addControl(new OpenLayers.Control.LayerSwitcher());
     map.zoomToExtent
     (new OpenLayers.Bounds(2500000,5000000,3100000,5500000));
   }
  </script>
 </head>
```

A `function init()` az oldal betöltésekor végrehajtandó függvény.

Most már van egy térképünk, mely az OpenStreetMap-et tartalmazza, és amelyen Bulgária ki van emelve az országhatár jelölése által. Az általunk a mapfile segítségével létrehozott különböző térképi rétegek megjeleníthetőségét kell még megoldanunk.

A különböző megjeleníthető tematikákat ki-be kapcsolhatóvá szeretnénk tenni úgy, hogy egyszerre egy tematika lehessen választható, mivel mindegyik tematikus réteg teljesen lefedi az ország területét. Ez fennáll azon tematikák esetében is, ahol diagramokkal ábrázolunk adatokat, mert ekkor a felület színezését is meghagyhatjuk a térképen. Bár a diagramjaink mérete jelzi a kerületre vonatkozó számérték nagyságát, a megjelenítendő térkép informatívabb, ha a kartogrammódszert és a diagrammódszert együtt alkalmazzuk.

Ahhoz, hogy a listából egyszerre csak egy tematikát lehessen választani, választógombot, vagyis úgynevezett *radio button*-t használunk. Egy választógomb létrehozása a szövegtörzsben történik, a következő módon:

```
<input type=radio name="layervalaszto" value="regions_museums"
      onclick="valt(this.value)"/>
```

Először megadjuk, hogy egy radion button-típusú elemet szeretnénk beágyazni, melynek neve `layervalaszto`. Az értéket a mapfileból vesszük – melynek elérési útvonalát a fejrészben rögzítettük –, és a mapfile-ban megadott rétegnév alapján hivatkozunk rá. Az `onclick` esemény pedig végrehajtja, hogy kattintásra egy bizonyos változó az általunk hivatkozott réteg értékét vegye fel és jelenítse meg. Ezt a változót (és függvényt) a html dokumentum fejrészében, a JavaScript kódban kell definiálnunk:

```
var aktReteg;
function valt(reteg)
   {
     aktReteg=reteg;
     wms.mergeNewParams({layers: [aktReteg,aktNyelv]});
      document.images.legend.src="http://mercator.elte.hu/cgi-
      bin/mapserv?map=/home/bscfold2007/vaaoaft/mapserver_data/bg3.map&mode
      =legend&layers="+aktReteg.replace(/,/g,'+');
   }
```

Ez a függvény teszi lehetővé, hogy az éppen kiválasztott rétegünk, valamint a hozzá tartozó jelmagyarázat jelenjen meg a térképen.

A jelmagyarázatra a szövegtörzsben a következőképpen hivatkozunk:

```
<img name="legend" src="http://mercator.elte.hu/cgi-
      bin/mapserv?map=/home/bscfold2007/vaaoaft/mapserver_data/bg3.map&mode
      =legend"/>
```

Egy image-típusú elemet hozunk létre `legend` névvel. Emellett megadjuk az elérési útvonalat, aminek a végén a `mode=legend` biztosítja, hogy a MapServer által előállított jelmagyarázatot kapjuk meg. Most látjuk, miért fontos, hogy a mapfile-ban adott rétegen belül milyen nevet adunk az egyes osztályoknak `(CLASS)`, ugyanis ezek a nevek jelennek meg a jelmagyarázatban is.

A mapfile-ban létrehoztunk két `annotation`, vagyis megírás típusú réteget; egyet a kerületek neveinek cirill, egyet pedig a latin betűs megírásukhoz. Azt is szeretnénk választhatóvá tenni, hogy melyik felirat legyen látható térképünkön. Ezt szintén választógomb segítségével tesszük meg:

```
<input type=radio name="labelvalaszto" value="regions_label_cyrillic"
      onclick="nyelvValt(this.value)"/>
```

A fent leírt rétegválasztóhoz hasonlóan, itt is egy `onclick` esemény nyomán egy változó felveszi az általunk választott réteg értékét. Ezt a változót (és függvényt) szintén definiálnunk kell:

```
var aktNyelv;
function nyelvValt(reteg)
   {
     aktNyelv=reteg;
     wms.mergeNewParams({layers: [aktReteg,aktNyelv]});
   }
```

V.2. A rétegek hozzáadása a weboldalhoz

Az oldal betöltésekor megjelenő térképünkön tehát látható az OpenStreetMap, illetve ezen az országhatár által kiemelve Bulgária. Láttuk, hogy hogyan tudunk más rétegeket ki-be kapcsolhatóvá tenni, illetve, hogy hogyan tudjuk a hozzájuk tartozó jelmagyarázatot megjeleníteni. Így ezek után már csak az a feladatunk, hogy a különböző rétegeket hozzáadjuk a weboldalhoz.

Elsőként a feliratválasztókat adjuk hozzá, elsőnek a latin betűs, majd a cirill neveket választó gombot. Mindkét esetben egy-egy réteg nevére kell hivatkoznunk a `value` megadásakor.

Ezek után következik két újabb választási lehetőség; az egyik Bulgária kerületeit és vízrajzát, a másik pedig a városokat tartalmazza. A vízrajz több különböző típusú réteg tartalmából tevődik össze, mert a tavak és a folyók különböző rétegen vannak, illetve a folyókból is kettő, egy vonal és egy sokszög típusú réteget hoztunk létre. A folyók esetében azonban a két réteget csoportosítottuk, így a kerületeket és a tavakat tartalmazó réteg mellett csupán ezt a csoportot kell meghívnunk és mindkét folyókat tartalmazó réteg látható lesz térképünkön.

A városok is csak egyetlen rétegen vannak, azonban a kerületeket itt is meg kell jelenítenünk, különben az OpenStreetMap tartalma nagyon összefolyna a városokkal, és a térkép olvashatatlanná válna.

A következő négy gomb segítségével a kultúrával kapcsolatos tematikák: a kiadott újságok, kiadott könyvek, múzeumok illetve múzeumi látogatások száma jeleníthető meg. Mind a négy esetben egy-egy sokszög típusú rétegre kell hivatkoznunk.

A választható tematikák másik csoportjában a lakosságra vonatkozó adatokat szeretnénk megjeleníteni. A termékenységi ráta, az élve születettek száma és a halálozási ráta esetében az előző négy réteghez hasonlóan csupán egy-egy réteget kell hozzáadnunk a térképhez. Az élve születettek, illetve a halálozási ráta nemek szerinti megoszlását kördiagrammal szemléltetjük. A diagramok mérete bár jelzi a kerületre vonatkozó számérték nagyságát, a megjelenítendő térkép informatívabb, ha a kartogrammódszert és a diagrammódszert együtt alkalmazzuk; így itt a `value` megadásakor két-két rétegre hivatkozunk.

A népesség számának és nemek szerinti megoszlásának ábrázolásához a mapfile-ban három réteget hoztunk létre: az egyik egy sokszög, a másik egy megírás, a harmadik pedig egy diagram típusú. Az első választható tematikánk, a népesség száma esetében a felületszínezést és a megírásokat jelenítjük meg, de mivel ezt a két réteget csoportosítottuk, csak a csoport nevére kell hivatkoznunk. A népesség nemek szerinti megoszlásának ábrázolása esetében e csoport neve mellett még a diagram típusú rétegre történő hivatkozás szükséges.

A két utolsó választási lehetőség esetében – megkötött házasságok, illetve válások száma – ugyanúgy járunk el, mint a népesség számának megjelenítése esetében, ugyanis a két sokszög típusú réteget itt is csoportosítottuk a hozzájuk tartozó megírásokat tartalmazó rétegekkel, így csak a csoport nevére kell hivatkoznunk.

V.3. Felmerülő problémák

Néhány dologgal csak a weboldalon történő megjelenítés után szembesülünk.

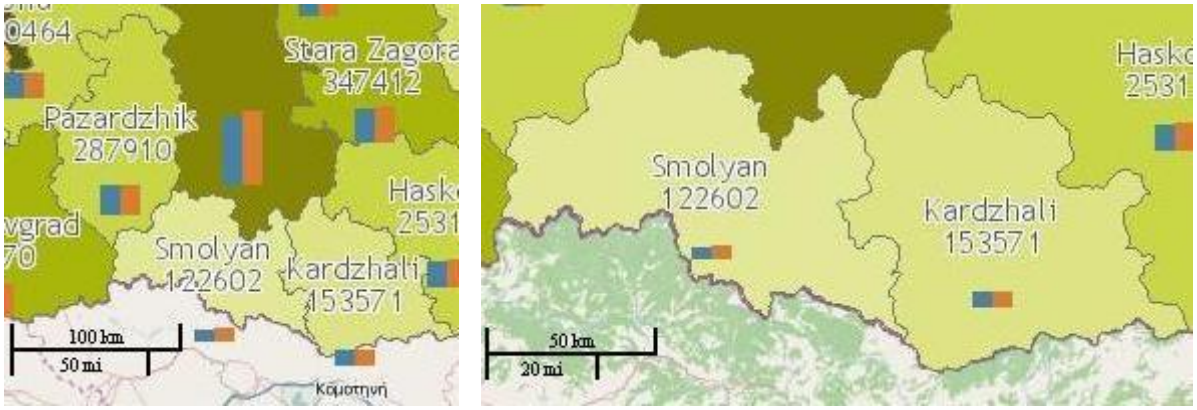
A feliratok színét úgy határoztuk meg, hogy a betű maga szürke (90 90 90), a kontúr pedig fehér (255 255 255), hogy minden színű háttér előtt jól olvashatók legyenek. A weboldalon történő megjelenítés után azonban látható, hogy a betűk kontúrja nem fehér, hanem színtelen, vagyis a kontúr helyett az OpenStreetMap tartalma látható és ez zavarja az olvashatóságot. Ez a probléma könnyen orvosolható azáltal, hogy a kontúrnak egy nagyon halvány szürke színt (254 254 254) adunk; így az fehérnek fog látszani a térképünkön.

18. ábra: *A betűk kontúrjának megadása fehér, illetve halványszürke színnel*

Azon tematikák esetében, amelyek nem csak felületszínezést, hanem kördiagramokat is tartalmaznak, megfigyelhetjük, hogy a kerületek nevének megírása éppen kitakarja a diagram lényegi részét. Ez azért van, mert a diagramok és a feliratok is annak a kerületnek a középpontjához igazodnak, amelyre vonatkoznak. Azonban a feliratok elhelyezkedése a középponthoz képest változtatható a mapfile-ban a `POSITION` paraméter segítségével. Lejjebb mozdítva a megírásokat a diagramok is olvashatóvá válnak.

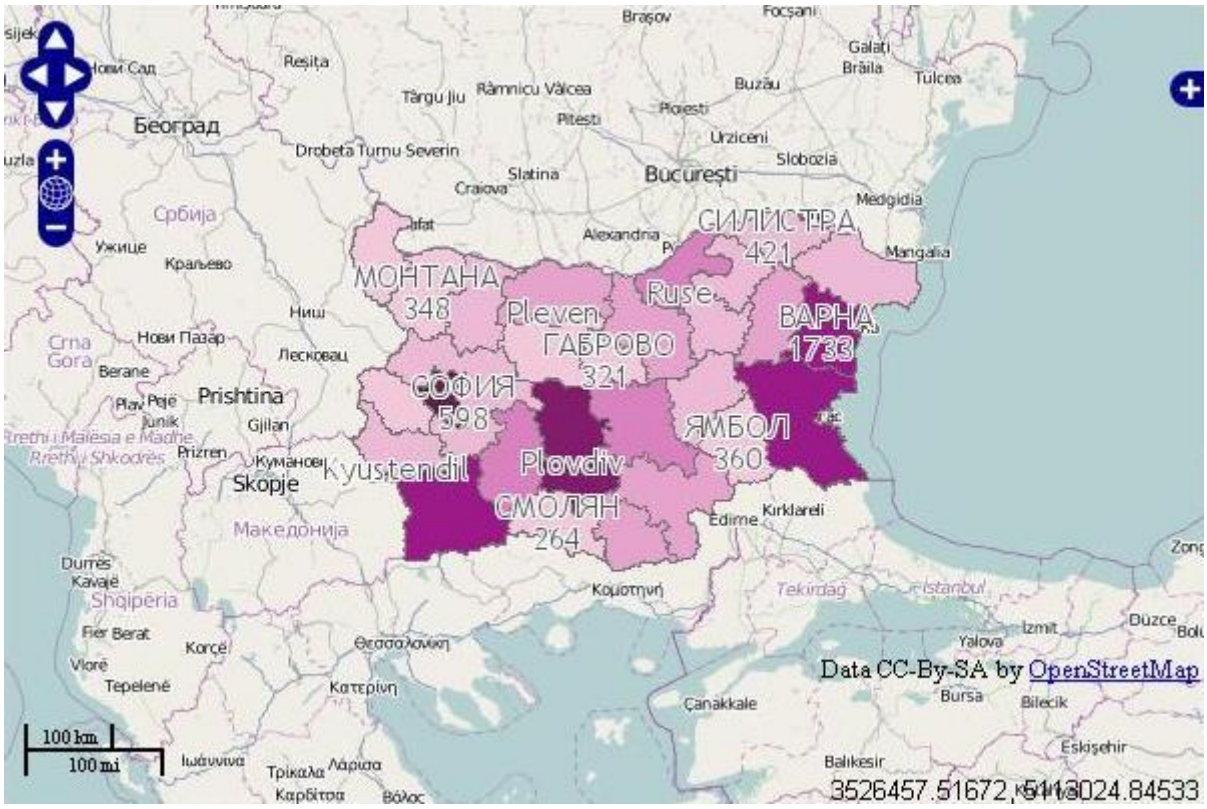
19. ábra: *A megírások a kerületek középpontjához igazítva, illetve eltolva*

Egyetlen rétegünk van, mely oszlopdiagramot tartalmaz. Az oszlopdiagram definiálásakor meghatároztunk egy maximális magasságot, mely a legnagyobb ábrázolandó értékhez tartozik. A MapServer ezt a maximális nagyságot veszi az összes diagram méretének, és ennek a középpontját igazítja a kerületek középpontjához. Emiatt, ha a térképbe nem nagyítunk bele eléggé, sok esetben a diagram a kerület területén kívülre esik. Itt a megoldás az lenne, ha a diagramok elhelyezkedését külön-külön tudnánk változtatni, de ez nem lehetséges. Egyetlen megoldás, hogy belenagyítunk a térképbe, így - mivel a diagram mérete nem változik – egy bizonyos nagyítási érték után a diagram vonatkozási helye egyértelművé válik.



20. ábra: *Az oszlopdiagramok elhelyezkedése nagyítás előtt és után*

Minden megírást tartalmazó réteg esetében igaz, hogy csak azok a feliratok jelennek meg, melyeknek van elég hely. A népesség, a megkötött házasságok és a válások számát ábrázoló tematikák esetében kettő megírást tartalmazó réteg is megjelenítésre kerül; az egyik amely a tematikához tartozik és a kerületek nevét, illetve a kerületekre vonatkozó számadatokat tartalmazza, a másik pedig a csak a kerületek nevét tartalmazó cirill betűs, vagy latin átírású neveket tartalmazó réteg. Az előbbi esetében a mapfile-ban a feliratok PRIORITY paraméterének értéke 10, vagyis azok fognak először megjelenni. Abban az esetben azonban, amikor nem nagyítunk bele a térképbe, előfordulhat, hogy az ehhez a réteghez tartozó felirat a mérete miatt nem fér ki, ellenben a csak a kerület nevét megjelenítő felirattal, amely így láthatóvá válik. Ekkor megtörténhet, hogy egyszerre cirill, illetve latin betűs megírás is látható lesz a térképünkön, attól függően, hogy éppen melyik feliratréteget választottuk. A megoldás az előbbi esethez hasonló; ha belenagyítunk a térképbe, megjelennek az adott kerületre vonatkozó, tematikus adatot hordozó megírások.



21. ábra: *A megírások megjelenése a nagyítás függvényében*

A rétegválasztó, illetve a feliratválasztó függvény definiálásánál látható, hogy minden esetben betöltésre kerül az `aktReteg`, illetve az `aktNyelv` változóhoz hozzárendelt réteg is. Ez azt jelenti, hogy akkor jelenik meg a térképünkön bármi – tematikus tartalom vagy megírás –, ha már kiválasztottuk a feliratok nevét és egy tematikát is. Amíg csak az egyik kiválasztása történt meg, egy nem megjeleníthető kép kis ikonja látszik a térképünk helyén. Ez úgy küszöbölhető ki, ha mind az `aktReteg`, mind az `aktNyelv` változónak már a fejrészben adunk egy értéket. Esetünkben:

```
var aktReteg="border", aktNyelv="regions_label_latin"
```

Ez azt jelenti, hogy ha először a feliratok nyelvét választjuk ki a weboldalon, akkor ahhoz megjelenik a határvonalat tartalmazó réteg. Ez azért jó, mert az már alapesetben is látszik, vagyis nem jelenik meg semmilyen tematika, amíg mi ki nem választunk egyet. Ha pedig először egy tematikát választunk ki, akkor az a latin betűs átírású feliratokkal fog megjelenni, de egy kattintással átválthatunk a cirill betűs feliratokra.

V.4. A weboldal

Az elkészült weboldalnak néhány egyszerű html-kód hozzáadásával szép megjelenítést biztosíthatunk.

Az oldal betöltésekor az OpenStreetMap látható, rajta kiemelve Bulgária; és a választható tematikák. Ha valamelyik tematikát, illetve megírást kiválasztjuk, akkor az megjelenik a térképen, valamint megjelenik a hozzá tartozó jelmagyarázat is

A térkép bal felső sarkában található vezérlők segítségével a térkép elmozdítható, illetve bele tudunk nagyítani. A jobb felső sarokban látható rétegválasztó segítségével ki tudjuk kapcsolni az összes Bulgáriára vonatkozó réteget.

22. ábra: *A weboldal képe használat közben*

VI. Összegzés

Diplomamunkámban bemutatásra került, hogy hogyan tudunk a MapServer segítségével adatbázisokból térképet előállítani és azt a weben megjeleníteni.

A dolgozat elején Bulgária statisztikai felosztását ismertettem. A rendelkezésemre álló adatokat a MapInfo Professional segítségével rendeztem adatbázisba, vagyis a különböző térképi objektumokat hordozó Shapefile-okhoz hozzákapcsoltam a bolgár Nemzeti Statisztikai Intézet honlapjáról .xls formátumban letölthető statisztikai adatokat.

Az adatbázis elkészítése után ismertettem a MapServert, illetve a mapfile felépítését. A mapfile-ban definiálható – a kartogrammódszert, illetve a diagrammódszert alkalmazó – tematikus tartalmat hordozó térképi rétegek készítésének általános leírása után elkészítettem a Bulgáriára vonatkozó térképi rétegeket leíró mapfile-t. Ez a tematikus rétegeken kívül tartalmaz általános térképi tartalmú, illetve csak megírásokat tartalmazó rétegeket is.

A térkép weben történő megjelenítéséhez az OpenLayers nyílt forráskódú függvénykönyvtárát használtam fel. A különböző változók és függvények fejrészben történő rögzítése után hozzáadtam a weboldalhoz a rétegek választhatóságát biztosító elemeket. Ismertettem, hogy a feltüntetni kívánt tartalom igénye szerint mely rétegeket kell egyszerre megjelenítenünk. A weboldal használata során felmerülő problémák kijavítása után elkészült egy weboldal, mely Bulgáriára vonatkozó tematikus térképeket mutat be.

Szakdolgozatomban csak egy töredékét mutatom be a MapServer biztosította lehetőségeknek, melyeknek száma igen nagy. A leírtak azonban alapot adhatnak az eszköz jobb megismeréséhez, új ötletekhez, akár szélesebb körben történő felhasználásához.

VII. Források

Könyvek:

ELEK István: Bevezetés a geoinformatikába; ELTE Eötvös Kiadó, Budapest, 2006

ELEK István: Térinformatikai gyakorlatok; ELTE Eötvös Kiadó, Budapest, 2007

Bill KROPLA: Beginning MapServer: Open Source GIS Development; Apress, Berkeley, 2005

Internetes források:

Bulgária közigazgatása:
http://hu.wikipedia.org/wiki/Bulg%C3%A1ria_k%C3%B6zigazgat%C3%A1si_beoszt%C3%A1sa

Nemzeti Statisztikai Intézet, Bulgária (НАЦИОНАЛЕН СТАТИСТИЧЕСКИ ИНСТИТУТ)
http://www.nsi.bg/index_en.htm

MapServer Documentation:
http://mapserver.org/documentation.html

MapServer és OpenLayers segédlet:
http://mercator.elte.hu/~saman/hu/

OpenLayers Documentation:
http://docs.openlayers.org/

OpenStreetMap:
http://www.openstreetmap.org/

Köszönetnyilvánítás

Köszönetet szeretnék mondani mindazoknak, akik segítséget nyújtottak e diplomamunka elkészítésében.

Elsősorban hálámat fejezem ki témavezetőmnek, Gede Mátyásnak, aki a témaválasztástól kezdve a dolgozat elkészültéig segítségemre volt. Végig biztatott, válaszolt kérdéseimre és utat mutatott a felmerülő problémák megoldása során.

Köszönöm Temenoujka Bandrova és Stefan Bonchev közreműködését, akik adatot szolgáltattak Bulgária térképeinek elkészítéséhez.

Köszönetet mondok továbbá a Térképtudományi és Geoinformatikai Tanszék dolgozóinak, illetve barátaimnak, akikhez bizalommal fordulhattam bármilyen kérdéssel.

És végül, de nem utolsó sorban, köszönöm családomnak, hogy biztosították a körülményeket a munkához, és végig támogattak.