SVEUČILIŠTE U SPLITU
**PRIRODOSLOVNO MATEMATIČKI FAKULTET**

ZAVRŠNI RAD

# “Grafička komponenta za mrežni prikaz znanja”

Vito Radovniković

Mentor: doc.dr.sc. Ani Grubišić

Split,2014

# Sadržaj

# 1. Uvod

U digitalnom svijetu čovjeku su sve informacije i podaci nadomak ruke. Najčešće nam podaci i informacije nisu baš u potpunosti razumljivi, a nekada ih uopće ne možemo percipirati, niti usporediti s prijašnjim znanjem. U ovom završnom radu, obradit ćemo problematiku grafičkog prikaza mrežnog znanja i definirati osnovne pojmove potrebne za izradu pripadajućeg programskog rješenja. Pripadajuće programsko rješenje pokušava pronaći jedan od načina prilagodbe i pojednostavljenja problema kojeg ćemo predstaviti u sljedećim poglavljima. Prikupit ćemo nekolicinu jednostavnih informacija, pokušati ih vizualizirati i prikazati krajnjem korisniku. Vizualizacija je dobar način učenja i pamćenja bitnih informacija. Razni načini vizualizacije olakšavaju memoriranje osnovnih pojmova kojih se lakše prisjetimo pomoću pripadajuće vizualne asocijacije. Danas sve više ljudi pokušava vizualnim putem upijati nova znanja i informacije. Davno je prošlo doba kada su ljudi samo čitali tekstove, pokušavali razaznati i riješiti probleme samo kroz obične rečenice. Ljudi su bića kojima je vizualna prezentacija od iznimne važnosti. Same vizualne prezentacije nam potpiruju maštu, jasnije predstavljaju određenu problematiku i jednostavno nas bolje pripremaju za upijanje novih informacija.

# 2. Uloga grafova u programiranju

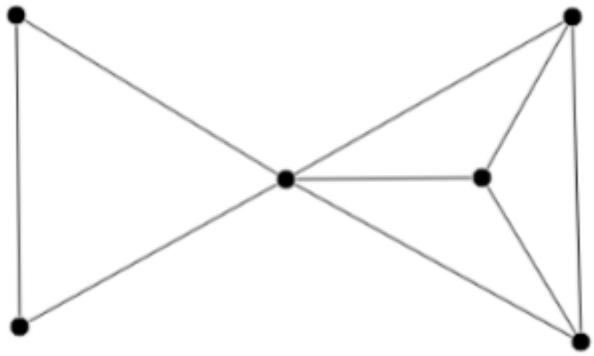
U matematici, odnosno kod teorije grafova, graf je predstavnik skupa objekata koji su povezani vezom. Grafovi su od iznimne važnosti u diskretnoj matematici i imaju veliku primjenu u svakodnevnom životu. Kod programiranja se također koriste neke od primjena grafova i njihovih teorija, pogotovo kod strukturiranja podataka te tokova algoritama. Postoji primjer crveno crnih stabala kao jednog od binarnih stabala koji se koriste kao podatkovna struktura za organiziranje dijelova usporedivih podataka. Kroz razne algoritme imamo i prikaz primjene grafova kao mjesta odluke i povezanosti tih odluka. Naprimjer, ako se radi i o algoritmu koji nam prikazuje dijeljenje cijelih brojeva, on mora imati na nekim mjestima odluke, odnosno račvanja gdje se pita da li je u pitanju cjelobrojno dijeljenje ili dijeljenje sa ostatkom, da li uopće može doći do dijeljenja i slično. Kod takvih pitanja i usmjeravanja također vidimo ulogu grafa i općenite povezanosti cijelog algoritma. Kod ovog projekta nećemo koristiti takvu primjenu grafova, već će grafovi biti više kao vizualno pomagalo.

## 2.1. Temeljni pojmovi iz teorije grafova

Na samom početku završnog rada, potrebno je definirati osnovne pojmove iz teorije grafova kako bismo mogli pristupiti rješavanju zadane problematike. Definirajmo prvo osnovni pojam grafa:

**Graf** je uređeni par G = (V, E) koji se sastoji od skupa čvorova (engl. Vertex ili Node) V = {v1, $v_2$, $v_3$, …,$v_n$} i skupa bridova (engl. Edge ili Line) E = {$e_1$, $e_2$, $e_3$, …,$e_n$} međusobno disjunktnih s V. Brid e spaja dva vrha u i v koje zovemo krajevi od e. Kažemo da je brid e incidentan s vrhovima u i v. Vrhove u i v zovemo susjednim vrhovima i pišemo e={u, v}. (Slika 1)

Bridove s barem jednim zajedničkim krajem nazivamo incidentni bridovi. **Brid** koji spaja isti par vrhova više puta nazivamo višestruki brid, brid koji spaja isti vrh sa samim sobom nazivamo petlja, a brid koji spaja dva međusobno različita vrha, nazivamo pravi brid ili karika. Graf je jednostavan ukoliko ne sadrži višestruke bridove i petlje. [ 1 ] [ 3 ]



Slika 2-Primjer grafa

Na temelju podataka o susjednim čvorovima i bridovima izrađujemo matricu susjedstva.

*Matrica susjedstva* (engl. Adjacency matrix) je jedan od najjednostavnijih načina za prikaz odnosa među čvorovima. Matrica informaciju prenosi tako da su redni brojevi čvorova ispisani horizontalno i vertikalno, a prelazeći kroz polja u matrici zapisujemo 1 ukoliko je došlo do podudaranja, a 0 ukoliko nema podudaranja među čvorovima. Detaljnije, pišemo:

$$A_{i,j}=\begin{cases} 1, & ako\ postoje\ bridovi\ među\ čvorovima\ i\ i\ j \\ 0, & ako\ ne\ postoje\ bridovi\ među\ čvorovima\ i\ i\ j \end{cases}$$

Matrica susjedstva često se koristi kao objekt u kojem se informacije o grafu spremaju u računalo.

*Planaran graf je svaki graf koji dozvoljava smještanje u ravninu $R^2$ tako da se njegovi bridovi sijeku samo u vrhovima tog grafa.*

**Šetnja** u grafu *G* je niz $W:=v_0e_1v_1e_2...e_kv_k$ čiji članovi su naizmjenično vrhovi $v_i$ i bridovi $e_i$, tako da su krajevi od $e_i$ vrhovi $vi-1$ i $vi$, $i=1,\ldots,k$. Kažemo da je *v0* početak a *vk* kraj šetnje *W*, da je *W* šetnja od *v0* do $v_k$ ili $(v_0, v_k)$-šetnja. Šetnja se, kada to ne umanjuje jasnoću, skraćeno zapisuje samo bridovima $W=e_1e_2...e_k$. Šetnja je zatvorena ako se početak i kraj podudaraju, tj. ako je $v0=vk$. Šetnju kojoj su svi bridovi međusobno različiti nazivamo **stazom**. **Put** je staza čiji su vrhovi međusobno različiti. Dva

vrha *u* i *v* grafa *G* su **povezana** ako postoji *(u,v)*-put u *G*. Graf je povezan ako su svaka dva njegova vrha povezana. [ 1 ] [ 4 ]

Kako bismo prikazali upotrebu grafova u stvarnom životu, obradit ćemo jednu poznatu temu, a to su šetnje koenigsberškim mostovima. (Slika 2)


Slika 3-Koenigsberški mostovi

Građani Koenigsberga (tada glavni grad pruskog kraljevstva, a poslije Kalinjingrad u Rusiji) zabavljali su se starim pitanjem mogu li prošetati svojim gradom tako da svaki od 7 mostova na rijeci Pregel pređu *točno jedanput* i završe šetnju u polaznoj točki. Leonhard Euler riješio je taj problem kao usputni zadatak na početku karijere vodećeg matematičara Ruske carske akademije u Petrogradu, gdje je 1733. godine dobio posao. Odgovor je bio nedvosmislen: takva šetnja nije moguća ako svaki dio kopna nije s ostalima povezan parnim brojem mostova. [ 4 ]

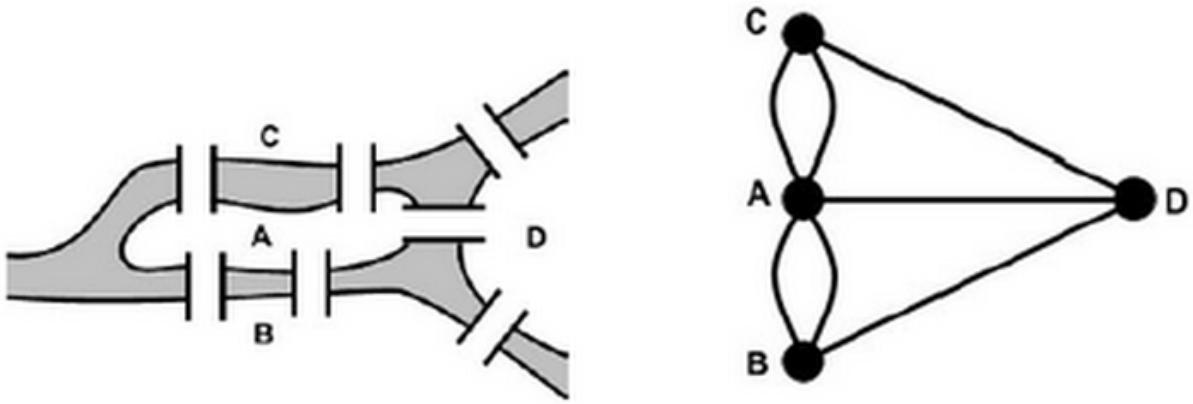
***Eulerova staza*** je staza koja sadržava svaki brid grafa (točno jedanput!). ***Eulerova tura*** u grafu je zatvorena Eulerova staza. Ako graf dopušta Eulerovu turu kažemo da je ***Eulerov graf***.

Odgovor na pitanje koji graf ima Eulerovu turu ili Eulerovu stazu daje sljedeći karakterizacijski teorem:

***Eulerov teorem**:* Povezani graf je Eulerov ako i samo ako mu je svaki vrh parnog stupnja. Povezani graf koji nije Eulerov sadržava Eulerovu stazu ako i samo ako ima točno dva vrha neparnog stupnja.

Graf s prethodne slike, koji predstavlja koenigsberške mostove, ima sva četiri vrha neparnog stupnja. Na temelju prethodnog teorema možemo ustvrditi da taj graf ne sadržava Eulerovu turu, ali ni Eulerovu stazu. Dakle, šetnja gradom koja prolazi svakim mostom točno jedanput nije moguća. (Slika 3)

Shematski prikaz nalazi se na slici ispod gdje su dijelovi kopna (obale B i C i otoci A i D) predstavljeni točkama a mostovi njihovim spojnicama, prototip je modela koji će odrediti definiciju pojma grafa. [ 1 ] [ 4 ]




Slika 4-Lijevo : graf čije su komponente mostovi i rijeka – Desno : shematski prikaz rasporeda elemenata grafa s lijeve slike

# 3. Vizualizacija znanja

U svakodnevnome govoru koristimo pojmove kao što su podatak, informacija i znanje. No što točno znače ti pojmovi? Podatak je slijed simbola i činjenica koje nemaju nikakav značaj. Podatak je vezan samo za sebe i nosi slijed nekakvih simbola i znamenki kao prikaz. Informacija je mnogo značajnija od podatka. Kada procesuiramo podatak tada mu pridajemo značaj i može nam odgovoriti na određena pitanja kao što su primjerice „tko?“, „što?“, „kada“ i „gdje?“. Informacija je upravo taj značaj koji pridjeljujemo podatku. Postoje razne informacije koje je jako teško razumjeti i usvojiti. Upravo zbog pojašnjavanja ovakvih podataka i informacija se koristi vizualizacija znanja.

Znanje je skup informacija koje su kognitivno procesuirane i integrirane u ljudski um. To je skup informacija unutar ljudske svijesti. Znanje je također dinamično jer se mijenja i razvija svakodnevno u ljudskoj interakciji sa okolinom. Vizualizacijom znanja se proučava vizualno prikazivanje određenih informacija na određenim medijima zbog lakšeg razumijevanja određenih informacija. [ 2 ]

## 3.1. Općenito

Vizualizacija se koristi za predočavanje najrazličitijih podataka, npr. statistike, rezultata mjerenja, podataka o računalnoj mreži i slično. Pritom vizualizacija pomaže lakšem poimanju podataka te često omogućava pronalaženje činjenica i zakonitosti koje bi bilo nemoguće uočiti pregledom podataka u numeričkom ili tekstualnom obliku. U medicini se koristi 3D vizualizacija rezultata ultrazvuka, magnetske rezonancije (engl. Magnetic Resonance Imaging) i računalne tomografije (engl. Computed Tomography, skr. CT) što je korisno za dijagnostiku i planiranje operacija. U kemiji i biologiji koristi se molekularna vizualizacija. Geografska vizualizacija ima široke primjene, od geologije i istraživanja nafte do planiranja cesta i ostale infrastrukture. [ 2 ]

Jedan od osnovnih ciljeva vizualizacije znanja je prijenos znanja gdje osoba koja ostvaruje interakciju s vizualnim medijem mora razumjeti ono što vidi. Ovaj prijenos je zapravo proces učenja pa je korisno uzeti u obzir i podatke od raznih psihologa i stručnjaka u obrazovanju. Mnoge teorije o učenju pomažu objasniti kako se znanje stvara iz informacija

i kako se taj proces zapravo bazira na socijalnoj interakciji sa drugima i sa okolinom. Postoje tri glavne teorije koje opisuju čitavi proces učenja: [ 2 ]

- *Biheviorizam (eng. Behaviorism)* – ova teorija dolazi od pretpostavke da se učenje bazira na čistim odgovorima i stimuliranjima odgovora bez obzira na mentalni model od kojeg se dobija informacija i način na koji on funkcionira ( tzv. „black box“ ). Kroz ovu teoriju se predlaže da se znanje rascjepi u male dijelove te ih se komad po komad uči.
- *Kognitivizam (eng. Cognitivism)* – je jedan korak ispred prijašnje teorije. Ovaj proces se većinom bavi dijelom procesuiranja podataka i informacija u znanje te kako to funkcionira unutar ljudskog uma.
- *Konstruktivizam (eng. Constructivism)* – teorija koja vidi proces učenja kao aktivan proces u kojemu osoba koja pokušava svladati nekakvo novo znanje konstruira ideje i koncepte bazirane na već usvojenom znanju. Odnosno, da bi ova teorija vrijedila, ispitanik već mora posjedovati nekakvo znanje i vlastita iskustva.

U popriličnom broju istraživanja do sada pokušavalo se pronaći idealan dizajn za prikaz informacija. Trenutačno postoje tri polja istraživanja koja pokušavaju pomoći izradi idealnog dizajna, no najbolje funkcioniraju kada se sva tri polja udruže. [ 2 ]

To su :

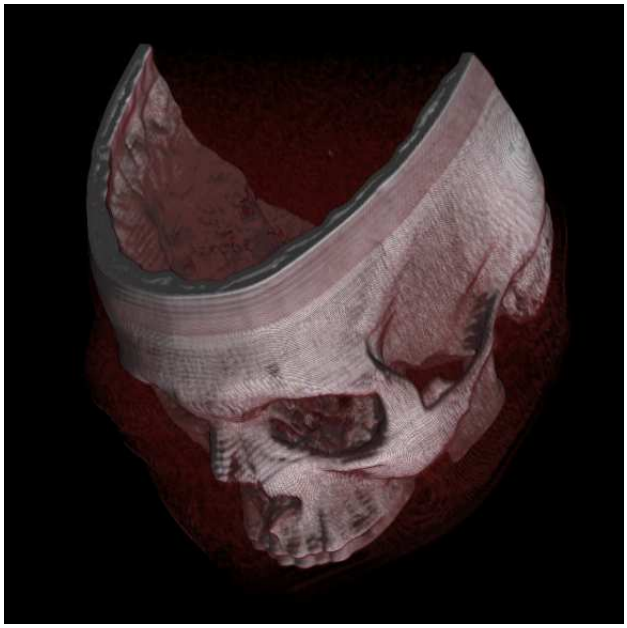
- *Dizajn informacija (eng.Information design)*– je dio znanosti koji se bavi pripremom informacija na način da su te informacije razumljive, lako dostupne i lako pretočene u akcije. Većinom se fokusiraju na statične vizualne forme kao što su mape i posteri.
- *Arhiktektura informacija (eng.Information Architecture)*– se koncentrira na grafički dio interakciji. Sučelja između ljudi i računala koja su većinom izgrađene vrhunskom arhitekturom, ali nemaju dobar dizajn niti prezentaciju.
- *Umjetnost informacija (eng.Information Art)* – se fokusira na estetski izgled i digitalnu prezentaciju prikaza informacija.

Kombiniranjem svih ovih polja, mogli bismo dobiti idealan format za prikaz informacija.

## 3.2. Vrste vizualizacije

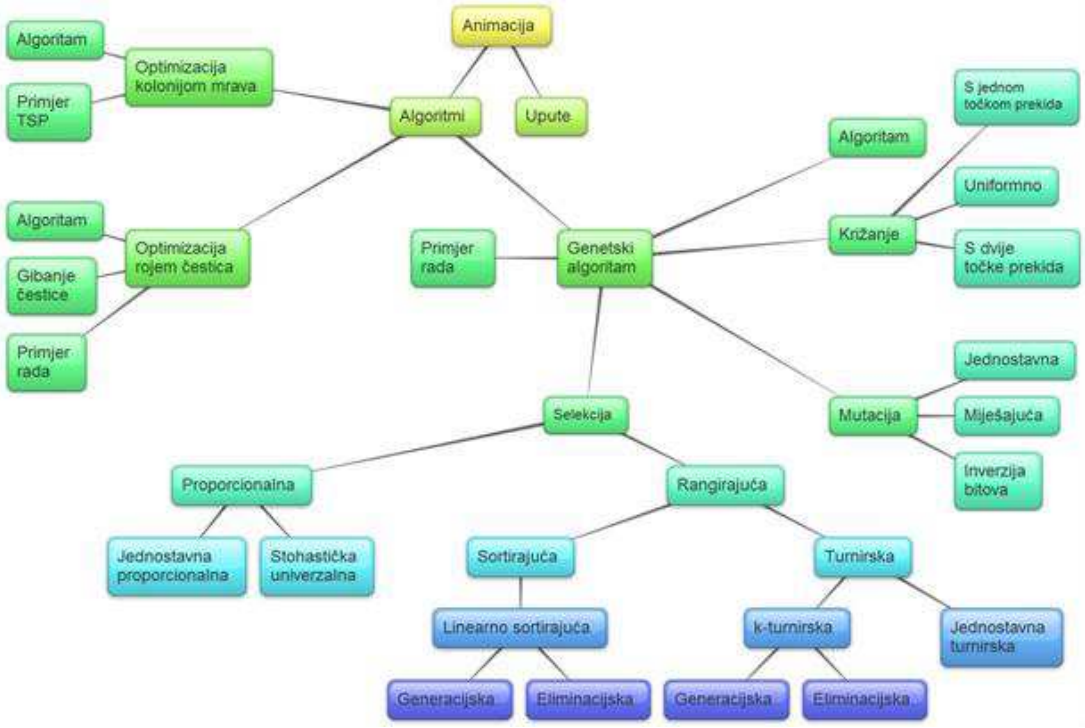
Glavna uloga vizualizacije znanja u računalstvu je prikaz podatak zbog lakšeg postavljanja hipoteza, zaključaka i donošenja odluka. Postoji mnogo različitih prikaza u mnogo različitih grana znanosti.[ 2 ]

- *Vizualizacija znanosti (engl. Scientific visualization )* – se sastoji od različitih metoda prikaza, odabira i transformacija podataka dobivenih kroz simulacije i eksperimente. Metodom vizualizacije ti podaci se mogu istraživati, analizirati , te uz razumijevanje može se doći i do novih zaključaka. Načini vizualizacije koji se koriste ovdje su većinom grafovi i animacije koji prikazuju dobijene podatke. Neke od tradicionalnih vizualizacija kod znanosti su vizualizacije korištene u medicinske svrhe, istraživanja kemikalija , fizike i mnogih drugih. Neke od metoda vizualizacije su takozvane **izopovršine** ( engl. Isosurface) koje se sastoje od trodimenzionalnih površina i funkcija koje reprezentiraju različita tijela te takozvani **Volume rendering** koji je dio vizualizacije koji se bavi prikazom 2D i 3D projekcija. (Slika 4)



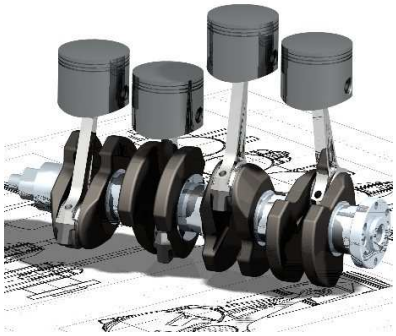
Slika 5-Volume rendering

- *Vizualizacija u obrazovanju* - U obrazovanju se vizualizacija, koristeći računalo, većinom bazira na izrađenim simulacijama koje se prikazuju studentima. Ovaj pristup je iznimno koristan kada je riječ o nekim stvarima koje su jako teško vidljive te zahtijevaju popriličan broj opreme da bi ih se proučavalo, npr. struktura atoma.
- *Vizualizacija znanja* - Vizualizacija znanja između dvije osobe se temelji na tome da se koriste razni vizualni formati kao što su naprimjer konceptualne mape (Slika 5), skice, dijagrami, slike i objekti u svrhu prijenosa znanja bez obzira koristilo se računalo ili ne.




Slika 6-Primjer konceptualne mape

- *Vizualizacija proizvoda* – Ova vizualizacija se temelji na tehnologiji koja prikazuje trodimenzionalne modele i tehnička crtanja modela proizvoda koji se mogu naći na tržištu. Ovaj proces je jedan od ključan kod proizvodnje različitih stvari kao što su automobili, zamrzivači i slično. No ova vizualizacija se također koristi i u drugim granama kao što su arhitektura i građevina. Tehnička crtanja su se prije obavljala rukom, no danas imamo napredna programerska rješenja za ovakve stvari, kao što je CAD ( engl. Computer aided design ). CAD koristi 3D modele, prototipove i razne simulacije za prezentiranje kranjeg proizvoda. (Slika 6)



Slika 7-Prikaz CAD dizajna

Interaktivne vizualizacije dopuštaju pristup, kontrolu, istraživanje, kombiniranje i manipuliranje različitim tipovima kompleksnih podataka, informacija i znanja.

Interaktivna vizualizacija je svaka vizualizacija koja dopušta interakciju korisnika s predmetima vizualizacije, pa tako korisnik može pomicati, okretati, dodavati nove ili brisati postojeće predmete unutar neke vizualizacije.

Postoji još mnogo primjena vizualizacija, ali mi smo pogledali neke od najčešćih koje se mogu susresti.

### 3.2.1. Mentalne mape

Tradicionalan način zapisivanja pomoću vođenja bilješki linearno, zapisivanje bilješki pomoću crtica (jedna ispod druge), često je monotono što podrazumijeva postojanje problema pri pamćenju zapisanih informacija. Rješenje ovog problema vidimo kroz korištenje mentalnih mapa koje olakšavaju učenje i pronalazak potrebnih informacija. U mentalnoj mapi informacije se prikazuju radijalno, a u izradi mape koriste se riječi i slike. Mentalna mapa predstavlja grafički prikaz gradiva (informacija) koje želimo organizirati i upamtiti. Mentalna mapa može imati stablastu strukturu koja omogućava hijerarhijski prikaz pripadajućih elemenata, a kako bi se poboljšao učinak mentalne mape, često se koriste jarke boje i sličice za stvaranje odgovarajućih asocijacija. Mentalna mapa je izraz našeg prirodnog razmišljanja te je zbog toga vrlo dobar alat u omogućavanju učenja. Pomoću mentalnih mapa svaka ideja se može „prikačiti“ za već postojeće elemente u mapi. Što je više takvih poveznica, lakše je upamtiti željenu informaciju. (Slika 7) [ 2 ]

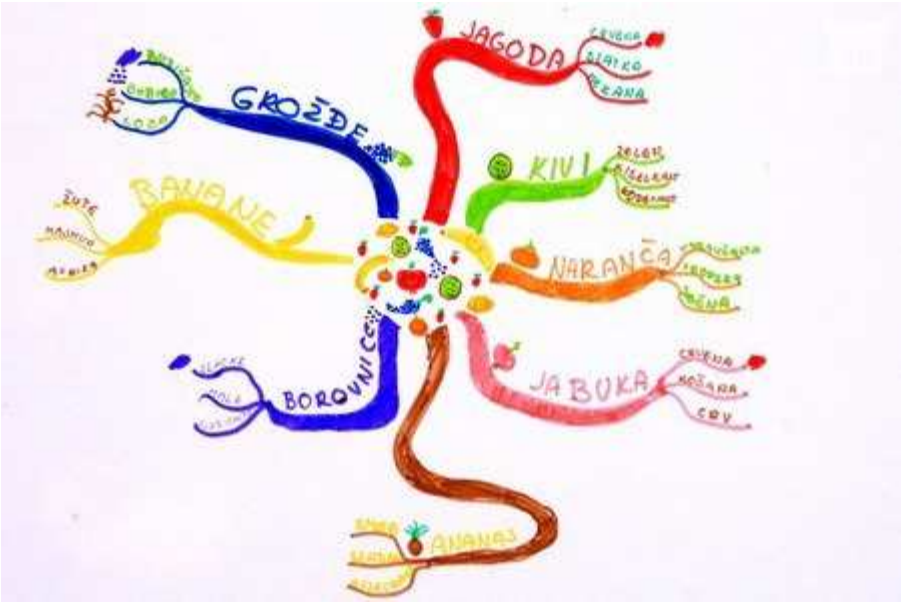
Prednosti mentalnih mapa:

- ne prikazuju samo činjenice, već i odnose među elementima koji olakšavaju pamćenje
- izrada mentalne mape zahtjeva dublju obradu informacija pa se prethodno moraju razumjeti odnosi među elementima
- izradom mentalne mape pratimo informacije na dva načina (vizualno i verbalno), pa ih se lakše prisjetiti nego informacija pohranjenih pomoću linearnih bilješki
- svaka mentalna mapa je jedinstvena, što pridonosi boljem pamćenju mape
- povećava koncentraciju prilikom učenja i pobuđuje kreativnost

Prije crtanja mentalne mape potrebno je pomno pročitati tekst koji želimo upamtiti. Na taj način dobivamo cjelovit uvid u gradivo koje želimo prikazati pomoću mentalne mape i

možemo ga jednostavnije e organizirati. Najbolje je početi s izradom na praznom papiru većeg formata, a u slučaju nestanka prostora, nalijepimo novi papir na postojeću listu. Poželjno je koristiti boje (3-4 boje kako ne bi bilo zbunjujuće) te crtati flomasterima jer je mapa tako jasnija i uočljivija, a njeni elementi „izlaze“ iz papira prema očima promatrača. U sredini papira crtamo središnji pojam. Takav početak omogućava širenje ideja u svim smjerovima. Slika treba biti dobra asocijacija na opću temu mape. Glavne ideje izlaze iz centra te se šire prema rubovima. Poželjno je za svaku granu prikazati vijugavim crtama kako bi se razbila monotonost, a slika postala privlačnija, upečatljivija i zanimljivija. Pisanje cijelih rečenica je nepotrebno te ruši koncepciju mentalne mape, bolje rješenje je pisanje ključnih riječi koje predstavljaju pojmove. Što više sličica – više asocijacija!

Neka znanstvena istraživanja o mentalnim mapama pokazala su da studenti lakše, kvalitetnije i efikasnije uče (do 15%) uz pomoć mapa nego pomoću klasičnih linearnih metoda. Danas, za takve i slične svrhe (poput pomoći u poslovanju), postoje i razne programske aplikacije pomoću kojih možemo izraditi mentalnu mapu.

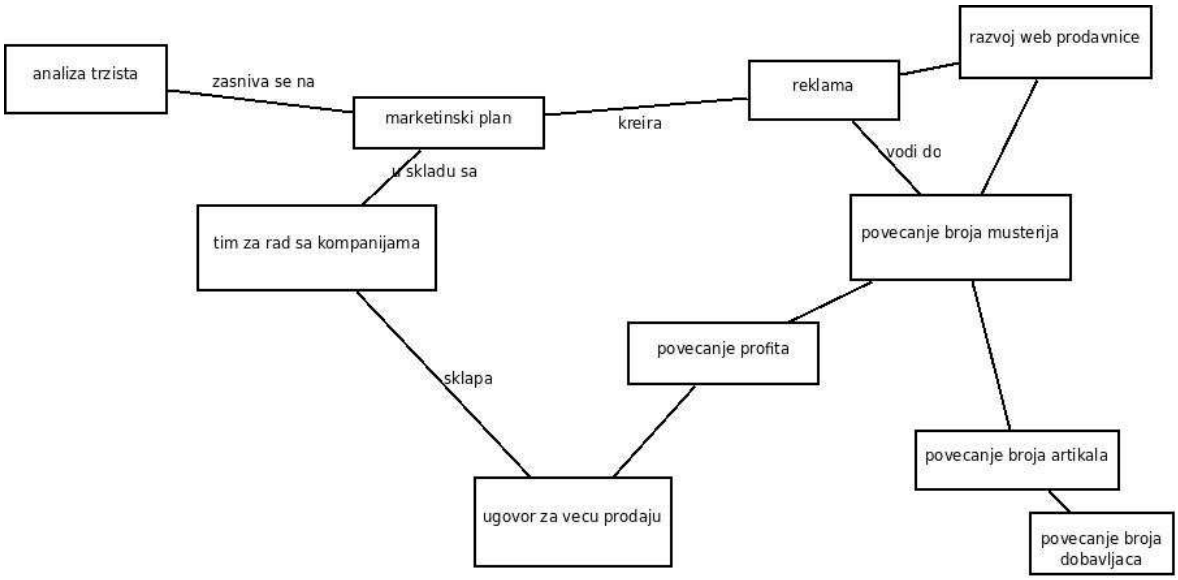



Slika 8-Jednostavna mentalna mapa o voću

Uz mentalne mape također postoje i konceptualne mape.U ovom završnom radu prikazat ćemo jednostavne konceptualne mape u kontekstu programiranja i rasporediti različite vrste elemenata te tako stvoriti pripadajuće asocijacije. [ 2 ]

### 3.2.2. Konceptualne mape

Konceptualne mape su mape koje se koriste kod vizualiziranja različitih veza između različitih pojmova. One su zapravo dijagrami koji olakšavaju nam prikaz raznih objekata i njihovih veza sa drugim različitim objektima. Sami objekti na konceptualnoj mapi su povezani strelicama i postavljeni u hijerarhijsku strukturu. Na samim strelicama također imamo i različita objašnjenja veza između različitih pojmova. (Slika 8)




Slika 9-Prikaz konceptualne mape

Samu tehniku konceptualnih mapa je razvio Joseph D. Novak. Novak tvrdi da konceptualne mape imaju svoje porijeklo u *Konstruktivizmu* gdje se smatra da sam učenik gradi svoje aktivno znanje. Sama mapa je način reprezentiranja veza između ideja, slika i riječi. Svaka fraza i riječ je povezana sa drugom riječi i ima svoj opis zbog čega je povezana. Konceptualne mape se koriste kao jedan od alata za razvijanje logike , općenite vještine učenja i mogučnosti upijanja novih ideja. Sami prikaz veza pomažu i objekata pomažu , poglavito učenicima, da vide kako individualna ideja je zapravo dio jedne cjeline koja ima puno veću svrhu.

Kada su konceptualne mape uvedene, primaran cilj im je bio da potpomažu znanstvenicima kroz radove te im olakšavaju zbunjujuće i opširne pojmove te njihove relacije sa drugim pojmovima.

Dobra konceptualna mapa ima svrhu da bude dio većeg konteksta koji pojašnjava neku cjelinu i odgovara na određena pitanja, dok mentalne mape su često samo razgranate mape koje pokazuju razne relacije jednog sa više pojmova. [ 6 ]

# 4. Programski alati

U ovom poglavlju definirat ćemo programske alate kojima se može vizualizirati znanje i definirati osnovne pojmove potrebne za razvijanje programske podrške različitih tipova vizualizacije. Upoznat ćemo se s razvojnim okruženjem u kojem ćemo razvijati pripadajuće programsko rješenje kao i s potrebnim bibliotekama implementiranih u odabrano razvojno okruženje. U današnje doba nam se nudi iznimno mnogo alata za dinamičan, interaktivan prikaz. Mnogo od tih alata je od iznimne važnosti u određenim granama znanosti i medicine, no mi ćemo se koristiti okruženjem Visual Studio i još nekim dodatnim bibliotekama.

## 4.1. Virtualni predmeti i razvojna okruženja

Kako bismo definirali pojam virtualnog okruženja, najprije ćemo definirati virtualni predmet. Virtualni predmet je svaki predmet definiran unutar memorije računala tako da ga računalo na zaslonu može prikazati korisniku uz mogućnost interakcije. [ 2 ] Definicija samog predmeta sastoji se od materijala od kojeg je napravljen i svojstava geometrije tog predmeta. Interaktivni prikaz podrazumijeva prikaz prilikom kojeg korisnik može u upravljati nekim parametrima prikaza u nekom realnom vremenu. Kao jednostavan primjer virtualnog predmeta uzet ćemo pravokutnik ( nećemo komplicirati trodimenzionalnim objektima). Pravokutnik može biti jednostavno definiran veličinom stranica i bojom ispune (npr. stranica a=10px, stranica b=12px  i boja ispune=žuta). Računalo može ovakav pravokutnik iscrtati na zaslonu, a korisnik pomoću miša ima mogućnost pomicanja virtualnog predmeta. Naravno, postoje daleko složeniji načini definiranja predmeta, no za naš projekt to ipak neće biti potrebno. Virtualni predmeti koje ćemo koristiti u projektu su: pravokutnici i dužine (linije) koje ih spajaju.

Postoje mnoga različita okruženja u kojemu se ovakav projekt može napraviti. Većinom sam razmatrao JavaScript sa raznim dodatnim bibliotekama kao što su KineticjJS i JQuery. Unutar KineticJS-a i Jquerya imamo mnoštvo pomoćnih metoda koje olakšavaju neke dijelove koda pa su bili ozbiljno razmatrani za ovakav projekt. Uz njih postojala je i mogućnost razrade projekta kroz XNA 4.0 , kao dio Visual Studia 2012. XNA je set alata koji se koriste za programiranje 2D i 3D igara, pa sam imao razne ideje implementacije

unutar XNA. No zbog mogućnosti prezentacije, lakše je bilo odabrati JavaScript i ASP.Net da izradu svega i lakšu prezentaciju.

Naposljetku,razvojno okruženje projekta završnog rada sastoji se od C map Toolsa i Visual Studia 2012(ASP NET i KineticJS).

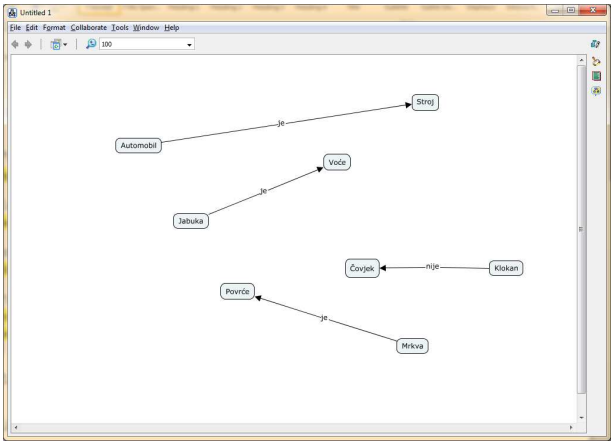
### 4.1.1. C map Tools

Ovaj alat omogućava korisnicima konstruiranje, navigaciju, dijeljenje i kritiziranje različitih modela znanja prikazanih preko konceptnih mapa. Kao što smo ranije navodili, postoji mnogi različiti načini prikazivanja vizualizacija znanja. Postoje mape, skice, 2D i 3D modeli, no CMAP Tools nam omogućuje iznimno intuitivnu izradu konceptnih mapa koje prikazuju neki od modela koje želimo vizualno prikazati. (Slika 9)

Prilikom izrade modela možemo birati koji su objekti povezani sa čime i na koji način, tako što nadodajemo opise na samo strelice koje pokazuju poveznice. Uz to , svaki od projekta možemo spremiti na različit broj načina.

Za potrebe ovog projekta, odabrao sam spremanje konceptne mape u oblike tekstualne datoteke.

Evo nekih od primjera uporabe CMAP Toolsa:




Slika 10-Cmap Tools

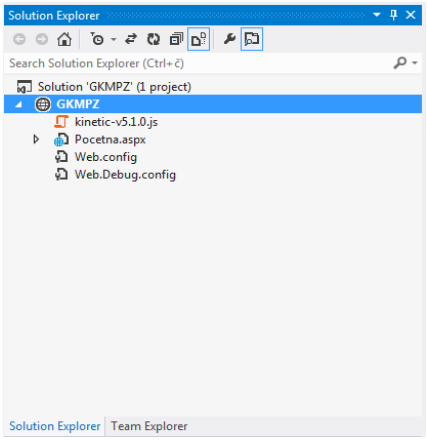
Nakon stvaranje nove CMAP mape dobijamo mogućnost dodavanja objekata, koji su u našem slučaju ovdje „automobil, stroj, jabuka, voće, mrkva, povrće, klokan, čovjek“ i dodavanja opisnih dijelova kao što su „je, nije“. Unutar CMAP toolsa može se stvarati i puno kompliciranije konceptne mape gdje bi imali više serijski povezanih objekata, no moj

projekt je dizajniran na način da postoji mogućnost međusobnog povezivanja dva različita objekta.

Nakon stvaranja konceptne mape, ona se može spremiti kao tekstualna datoteka ( File->Export CMAP as->CMAP Outline ) . Nakon spremanja te mape, moj dio projekta napravljen u okruženju Visual Studio 2012, učitava tu datoteku i manipulira podacima.

### 4.1.2. Visual Studio 2012

Kroz okruženje Visual Studio 2012 stvorio sam webstranicu i koristio neke ugrađene metode i funkcionalnosti samog ASP Net-a. No uz to sve bilo je potrebno i grafički predstaviti podatke dobijene kroz tekstualnu datoteku, a za izradu tog dijela projekta sam odlučio ugraditi KineticJS. (Slika 110) KineticJS je biblioteka koja spada u JavaScript koja sadrži mnoštvo metoda i proširenja za grafičke prikaze unutar HTML-a 5. Unaprjeđenja samog 2D modela i mnoštvo metoda koje sadrže, poprilično olakšavaju posao.

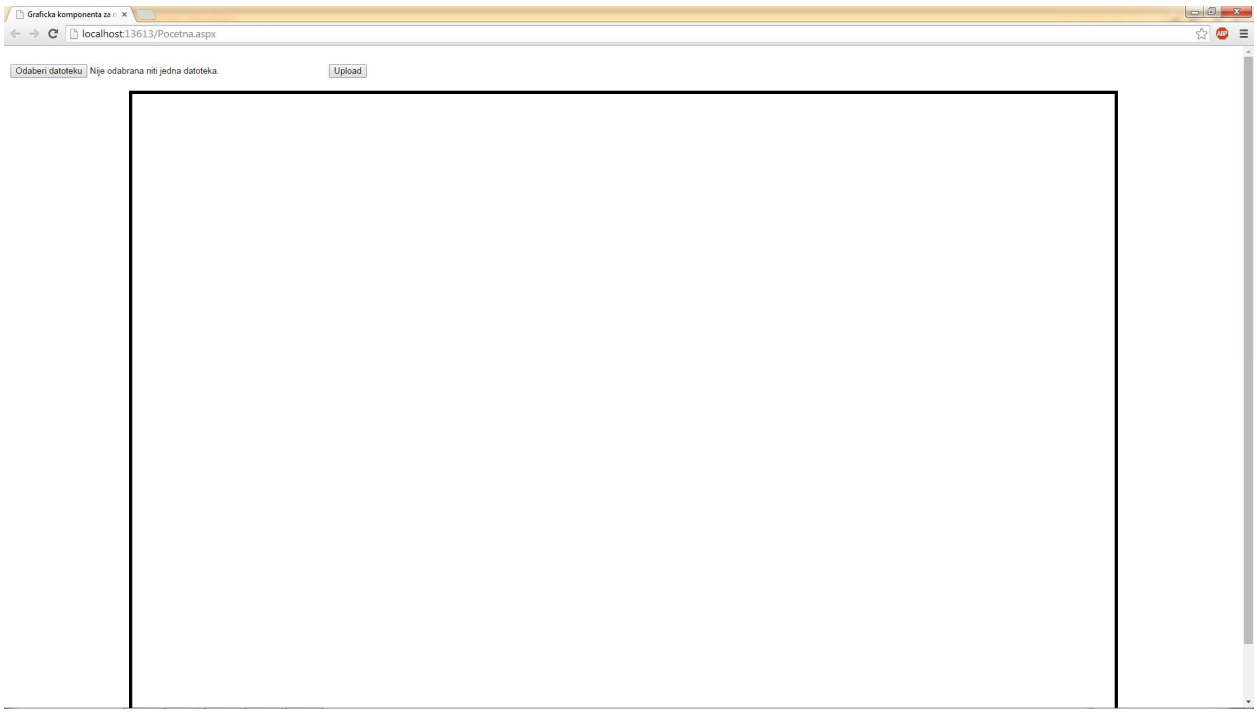



Slika 12-Prikaz stranica i biblioteka

# 5. Programsko rješenje

Kao što i sam naziv projekta nam kaže, cilj samog programskog rješenja je da imamo grafičku komponentu koja nam može prikazivati znanje kao mrežu. Od stvaranje konceptualne mape kroz CMAP Tools, učitavanje tekstualnog dijela i prikazivanja ga u smislu mreže.

Sami izgled aplikacije je poprilično jednostavan i lagan za korištenje. (Slika 131)



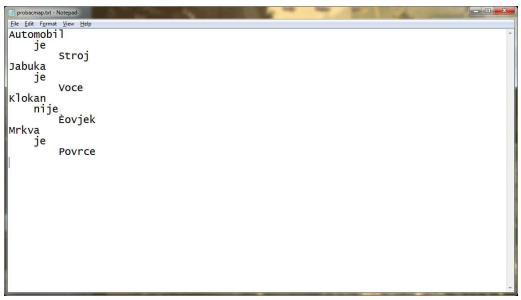

Slika 14-Izgled sučelja

U gornjem uglu imamo mogućnost odabiranja datoteke sa mrežnog diska i njeno učitavanje kroz program. Kada odaberemo samu datoteku i pritisnemo dugme za učitavanje ( „Upload“ ) . Dio koda za iscrtavanje preuzima rezultate algoritama i crta pravokutnike i linije između njih. Svaki od pravokutnika se može povlačiti i postavljati na druga mjesta, a linije se ponovno iscrtavaju sa svakom promjenom pozicije pravokutnika.

## 5.1. Zadaci

Popriličan broj malih zadataka je morao biti riješen da bi sve djelovalo kao cjelina. Prvo je bilo potrebno učitati samu tekstualnu datoteku. (Slika 152) Prilikom učitavanja se koristio

StreamReader objekt i njegove metode su nam omogućile čitanje cijele tekstualne datoteke.




Slika 16-Prikaz tekstualne datoteke

Nakon što smo pročitali samu datoteku, pridijelili smo svaki od redaka u zasebno mjesto u nizu. Tada je taj niz bilo potrebno poslati sa aspx stranice sve do JavaScript koda. To se postiglo korištenjem JavaScript Serializer objekta koji nam je omogućavao prijenos podataka.

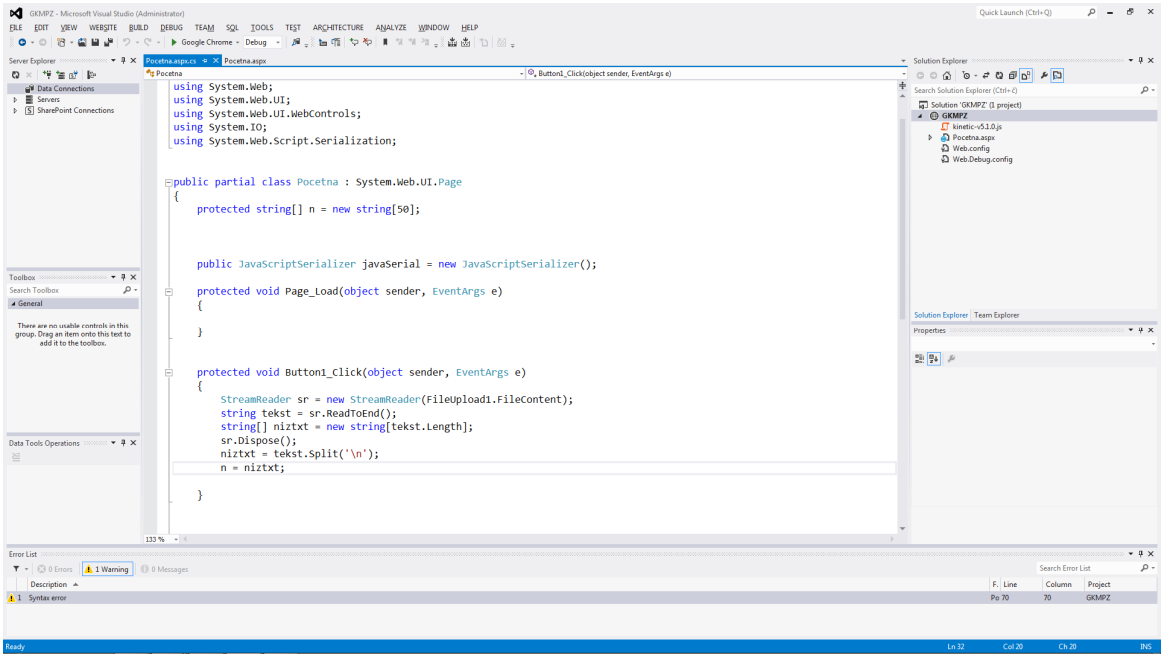
Nakon što su podaci uspješno dobavljeni kroz JavaScript kod, potrebno je odvojiti imenice i glagole, te ih pridijeliti vlastitim nizovima te ih prikazati na ekranu.

Prikazivanje pravokutnika i imenica nije dovoljno, već moramo stvoriti i poveznice između pravokutnika te im dodati glagole koji su im pridijeljeni.

Ovo su neke od glavnih zadataka koji su morali biti ostvareni da bi funkcionirao čitavi projekt, no naravno, svaki od ovih zadataka ima i svoje podzadatke koji su podrobnije opisani u seminaru završnog rada.

## 5.2. Implementacija

U ovom poglavlju ćemo prikazati neke od implementacija određenih zadataka. Prvo je potrebno učitati samu tekstualnu datoteku i pročitati njen sadržaj. (Slika 173)




Slika 18- Izgled dijela koda u Pocetna.aspx.cs

Kao što je već navedeno, to smo postigli tako što smo stvorili novi objekt StreamReader i uz pomoć njegovih metoda , sadržaj tekstualne datoteke smo postavili u varijablu „tekst“ koja je onda svoj sadržaj rascijepila i postavila u pomoćni niz „nizxt“ i na kraju je taj niz postavljen u zaštićeni niz ( koji je veličine 50 ) „n“. Zbog ovakvog limita projekt neće biti funkcionalan za konceptualne mape čije bi tekstualne datoteke bile veće od 50 redova.

Nakon pridjeljivanja tekstualne vrijednosti u niz, potrebno je i dobaviti sam tekst kroz JavaScript. [ 5 ] (Slika 194)

```
//Prijenos podataka sa inputa.
var nizteksta = <%= this.javaSerial.Serialize(this.n) %> ;
```

Slika 20- Kod za prijenos niza u JavaScript

Nakon prijenosa podataka prelazimo u dio u kojemu putem KineticJS-a upravljamo dobijenim podacima.

Prvo stvaramo pozornicu (engl. stage ) koju postavljamo unutar „containera“ koji se nalazi unutar crnog obruba. U sklopu ove pozornice ćemo stvoriti slojeve koji će sadržavati pravokutnike, tekst i linije koje ih povezuju. Svaki od objekata će biti moguće micat osim linija koje će biti povezane sa pripadajućim pravokutnicima.

Nakon pozornice stvoriti slojeve (engl.layer) na koje crtamo objekte. Tekst i njemu pripadajući pravokutnik će biti u istome sloju tako da se skupa mogu povlačiti bilo gdje .

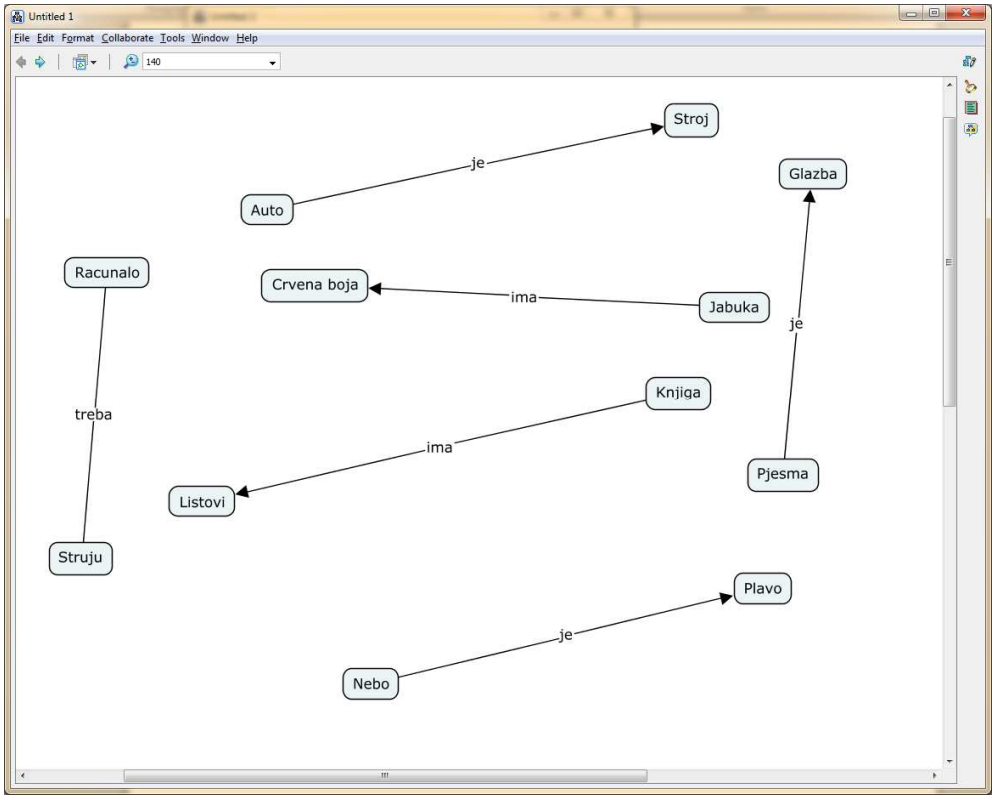
Stvaranjem pozornice i slojeva imamo grafičke predispozicije za postavljanje objekata za prikaz korisniku. Nakon stvaranja navedenog, koristimo algoritme koji odvajaju imenice i glagole te ih postavljaju u pripadajuće nizove. [ 5 ]

Tada crtamo pravokutnike i pridjeljujemo im tekst odvojen iz tekstualnog prijenosa. Nakon toga iscrtavamo linije između samih pravokutnika.

Uz to, na neki način moramo i postaviti glagole koji opisuju odnose, a njih nam je najlakše postaviti malo dalje od samih pravokutnika, no dovoljno blizu linijama jer nismo u mogućnosti iscrtavati tekst iznad same linije kroz KineticJS.

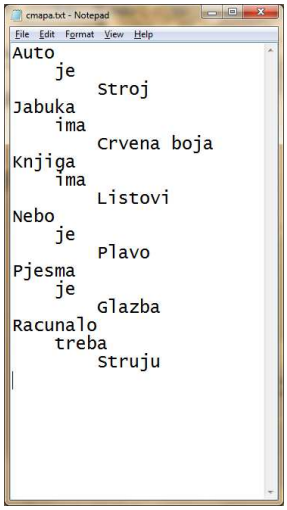
## 5.3. Prikaz rada

Da bi se dobro prikazao i objasnio sam način rada, najbolje je prikazati par slika i pojašnjenja koja dolaze uz njih. Da bi sam projekt funkcionirao potreban nam je alat CMAP Tools. U njemu prvo stvaramo kognitivnu mapu sa objektima koje mi želimo.




Slika 21-Prikaz konceptualne mape napravljene za testiranje

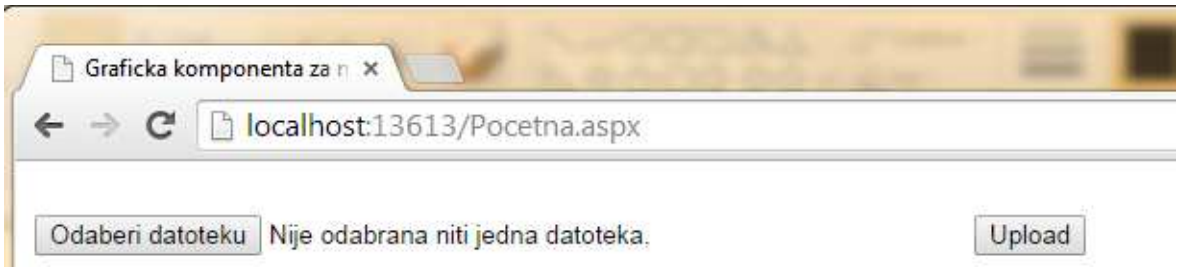
Nakon stvaranja konceptualne mape (Slika 225) sa objektima koje želimo unijeti, potrebno je spremiti samu mapu kao tekstualnu datoteku. To možemo učiniti preko opcija „File->Export Cmap As->Cmap Outline“.




Slika 23-Tekstualna datoteka napravljena za testiranje

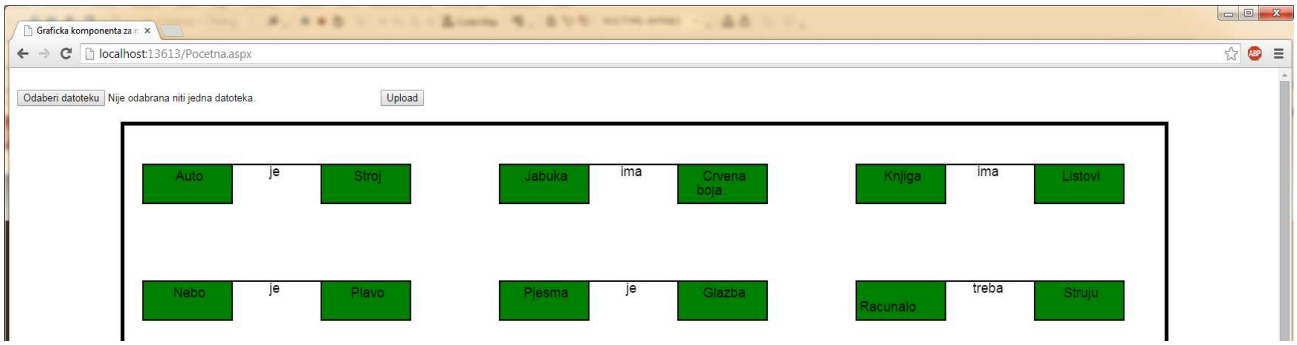
Nakon što imamo potrebne podatke u potrebnom formatu (Slika 246), možemo pokrenuti grafičku komponentu i učitati potrebne podatke. Također ne smijemo previdjeti činjenicu da CMAP tijekom stvaranja tekstualne datoteke nadodaje jedan prazni red na samome kraju. Ne smijemo dopustiti da nam to utječe na samo učitavanje datoteke!

Nakon što pokrenemo sam projekt, biramo tekstualnu datoteku koju želimo učitati. (Slika 17)




Slika 25-Prikaz dijela za učitavanje datoteke

Nakon što smo odabrali tekstualnu datoteku i učitali je, pritiskom na dugme „Upload“ , na dijelu „containera“ niže nam se iscrtava prikaz kako su objekti povezani.




Slika 18 – Izgled iscrtavanja

Ako želimo, možemo odabrati odmah i drugu datoteke tekstualnu, pod uvjetom da je izrađena kroz CMAP tools, i odmah će se iscrtati  čitava slika. (Slika 268)

# 6. Zaključak

U ovom završnom radu obradili smo problem vizualizacije mrežnog prikaza znanja i objasnili osnovne pojmove potrebne za definiranje prethodno spomenute problematike. Cilj pripadajućeg programskog rješenja je prikazati podatke dobivene iz tekstualnih datoteka kao grafičke komponente mreže koja omogućava lakše razumijevanje i percepciju znanja za krajnje korisnike. Način korištenja nije ograničen na unos podataka samo kroz CMAP tools, već se podaci mogu unijeti pomoću bilo kakve tekstualne datoteke koja svoje podatke zapisuje na isti način kao što ih i CMAP zapisuje. Potrebno je da je svaki objekt u vlastitom redu i da su dva povezana objekta odvojena glagolom koji ih povezuje. Pripadajuće programsko rješenje ima svoje nedostatke i ograničenja, naravno. Jedan od glavnih nedostataka jest nemogućnost čitanja tekstualnih datoteka koje imaju povezane više od dva objekta. To se može razriješiti osmišljavanjem boljeg algoritma za samo učitavanje i odvajanje podataka. Uz to, jedna od dodatnih stvari koja se može implementirati je da se na samo odabiranje pravokutnika ispisuju svi pravokutnici i njihov tekst povezan za njih. Postoji mnoštvo načina kojima se rješenje može poboljšati i unaprijediti. Ipak, prethodno opisano programsko rješenje kao cilj postavlja prikazivanje jednostavnih informacija koje su međusobno povezane, tako da ih predstavi kao oblik znanja koje možemo dobiti. Upravo novim znanjima možemo poboljšati ovo programsko rješenje te ne smijemo zaboraviti da se ipak radi o novom području istraživanja koje će se u budućnosti zasigurno značajno razviti.

# 7. Literatura

[1] Darko Veljan : „*Kombinatorna i diskretna matematika*“, Algoritam , 2001. godina.

[2]Robert Meyer,T.Smith,A.Mitchell*:*“*Cognitive Structure of Episodic Knowledge*“, University of Ohio, 1982. godina.

[3] Newman M.E.J. : „*Networks: An Introduction*“, Oxford University Press, 2010. godina.

[4] Newman M.E.J.,Albert-Laszlo Barbasi,Duncan J.Watts: „*The Structure and dynamics of Networks*“, Princeton University Press, 2006. godina.

[5]Mark Pilgrim: „*O'Reilly HTML5: Up And Running*“, O'Reilly Media , 2010. godina.

**[**6] Novak D. J.: „*Learning, Creating and Using Knowledge: Concept maps as facilitative tools for schools and corporations*“ , Taylor & Francis, 2010. godina.